1. Introduction

This is an introductory tutorial that outlines the basic usage of Yafaray and its workflow differences in comparison with Blender internal renderer. Through analyzing a simple scene we will learn basic stuff about creating right materials, setting up lights and rendering with Yafaray. In this tutorial it will be assumed that you have Blender and Yafaray already installed, know your way around Blender user interface and have basic understanding of CG.

**Note:** The Yafaray version used for this tutorial is from the experimental branch, which differs from the official stable release 0.1.1. The biggest visual difference is in the exporter GUI with additional features and different layout, but the buttons and toggles are named the same, so I hope this won't be a big problem. I choose to reference the new version because 0.1.1 is old and hopefully will be replaced soon.

I have prepared a simple scene, which it will be used to explain things. So, [download] it and fire up Blender. The scene is already complete, with material lights and render settings. The downloaded archive also contains all the needed textures on a relative path ./maps, so no unpacking is needed.

2. Lights & preview render

2.a Intro

The first thing you need to know is that Yafaray reads and uses settings from Blender GUI, but not all of them, so not all of Blender features are supported. What are those supported settings and features is neatly described in the [User guide](#), so you should always have the user guide at your reach (I still do :). The features distinct to Yafaray are controlled through the Yafaray export script. To add confusion some of the settings for a particular object (say lights) are read from Blender and some are set in the exporter script. You should make yourself familiar which are those settings.

The tutorial scene already has a script window opened on the right. If Yafaray GUI is not showing (as will be the case if you use 0.1.1 version) then navigate to scripts->render->Yafaray Export <version>. The GUI should pop up. If not, check the [Installation page](#).

I planned this tutorial to be an 'workflow explanation' so the scene you downloaded is already complete. I will explain the motivation behind the techniques I used to create it.

2.b About GI

You probably choose Yafaray because you want your scene to look more realistic, that means you want to take advantage of Global Illumination (or Indirect illumination) features that Yafaray simulates. Yafaray offers three GI lighting methods – Photon mapping, Pathtracing and Bidirectional pathtracing. Out of these, photon mapping (PM) is the most widely used, because is the most flexible and offers the best speed/quality ratio. PM is especially good at lighting closed spaces (i.e. areas where the background and/or light sources are mostly blocked by geometry).

The GI effects that Yafaray is able to simulate are:

- **indirect lighting**: transmission of illumination between surfaces
- **color bleeding**: transmission of color between adjacent surfaces
- **caustics**: light concentration as a consequence of reflective or refractive material
Note: To keep things shorter, Global illumination methods will be covered in an eventual future tutorial. In this tutorial we will be using only Direct lighting method.

See also: Lighting methods

2.c Setup lighting method for previews

First let's setup a lighting method for our test render. We will use the Direct lighting (DL) method for this. DL is fast and thus good for previews, but can also be a choice for finals – especially on simpler scenes, where light bouncing does not contribute much to the lighting and in animation, where render speed is a huge factor. DL does not calculate light bouncing but it does support Ambient occlusion (AO) – a fake approximation of GI, which renders much faster than the current Blender 2.4 AO implementation. DL method can also render caustics.

Lighting methods are set in the Settings tab of the export script. There you have a drop down with render presets you can create and choose from. Select the already created 'preview' preset from the drop down. You can choose the method of lighting from the Method of lighting dropdown, but leave it set to DL.

![Settings Tab](image)

DL has only a few specific settings, which are:

- **Use caustics:** It enables simulating caustics. We won't need to enable it in this tutorial
- **Use AO:** This enables Ambient occlusion. There is not much point in not using AO, so this toggle should be always enabled.
- **AO samples:** How detailed is AO. More samples means that occluded areas are more smooth, but also more render time. 16 – 32 should be enough for previews and middle-quality finals.
- **AO distance:** How distant the algorithm searches the scene for geometry that will occlude the currently rendered point. The bigger the number the more accurate the occlusion and the more render time. 1.0 is usually a good choice, but this depends on the scale of your scene (see note).
- **AO color:** The power or 'brightness' of AO.

Note: Numerical settings in Yafaray are directly related to Blender units. So for example if AO distance is set to 1.0, that means that the occlusion rays will be 1 Blender unit (BU) long. This relation has a lot of important implications, some of which will be considered later.

At this point, before we render anything, I must tell a few things about lights, so the next subsection covers light creation and settings.
See also: Ambient Occlusion

2.d Lights

The basic workflow for setting lights is: you create a light (Yafaray understands all Blender lights, except hemisphere), you tweak the options in Blender (if the light has any options there that relate to Yafaray) and then you set specific options in the Objects tab of the export script. At first when you create the light, it's not set in the export script, so you must navigate to the objects tab and select the subtype of the light (even if the light has only one subtype, you must still select something from the dropdown).

I will only list Blender lights and related subtypes in Yafaray and leave their feature description to the user guide:

- Sun → Sun, Directional
- Area → Area
- Spot → Spot, IES Light
- Lamp → Point, Sphere

Yafaray has also a meshlight, ie. every mesh can emit light. That is also set in the objects tab, when a mesh object is selected in Blender.

Note: The objects tab GUI changes depending of what you have selected in the Blender object window.

See also: Lights, Meshlight

2.e Clay render

For our first render of the scene we will use the Clay render option in the settings tab. When enabled, this option overrides all your materials with a default white shiny diffuse and such is useful for:

- fast previews of scene layout & geometry
- testing lights setup & illumination contribution of specific lights
- as a render pass that serves for composition techniques with GI renders

See also: Clay render tutorial, Compositing tutorial

We will use the DarTide Sunsky background model (world tab in the export script), which is a good background for a lot of scenarios. For a more throughout explanation of backgrounds see Background settings.

We will use the 'late afternoon' light rig on layer 20. That rig uses 3 lights: a key sun light, two area fill lights and a spot fill light. Here I will talk about the motivation behind such setup and it's use. We will render the scene three times, each time enabling a light to see it's contribution. First render only with sun lamp enabled:
The sunlamp is used as a key light (more about key & fill lights in: Studio lighting tutorial), meaning that this light contributes the most lighting on the scene. It's color is that of a late afternoon sun, so from yellow toward orange and red, and it's shining from a low angle to produce long shadows. The sun lamp can produce sharp or soft shadows, depending on the angle parameter – the lower, the more sharp are the shadows. The sun lamp is used on open scenes, when well defined parallel shadows are needed. In the above clay render we clearly see how the sunlamp is tinting the objects orange.

See also: Sun light

Next enable the area lights and re-render:

The area lights are used as fill lights, meaning they help the key light to illuminate the scene more
evenly. In our case the area lights represents the skylight - the scattering of light from the atmosphere – and they have a bluish tint, to bring the scene into a more neutral color balance.

Area lights produce soft shadows and are useful in many situations as fill or key lights. They should be shaped in accord to the light source they represent (a window frame, for example) and should not be used in big numbers, as they are slower to render than other types.

Sunlamp and arealights have also a **samples** parameter, that is used to reduce noise. The higher the samples, the more noiseless and smooth the illumination will be, but that will also result in longer render times. An amount of 16 - 32 samples is a good speed / quality compromise.

**See also:** [Area light](#)

Finally the subtle spot lamp brightens the dark areas a little more, while still preserving the form of the vases:

![sunlamp + arealights + spotlight](image)

Spotlights are very useful to lighten areas that does not receive enough light from your main lights, although that is not their only use. They are also very fast to render. To limit the sharpness of their shadows, a **soft shadows** option has been introduced in version 0.1.2. In the example we used soft shadows with the highest possible fuzziness level, to not create any visible and well defined shadows coming from the right, as that would look wrong on our open scene. The **samples** parameter in this case controls the smoothness of the 'penumbra' region of the shadow.

**See also:** [Spotlight](#)

To show the use of area lights as key lights, there's another light rig on layer 19. This rig could simulate a partially cloudy sky where the incoming light is very scattered. Note the soft & not well defined shadows that the area light produces.
3. Materials

Migrating from the Blender material mindset to Yafaray's is probably the toughest thing that beginners have to adapt to. Yafaray material creation workflow is different than in Blender. While in Blender every material has the same set of settings, Yafaray has its materials strictly defined by shader types.

*Note: material and shader are the same thing – both means *ways to describe how an object reacts to light*. It's just that programmers and people with software that allows you to write your own algorithms, use the word shader, while artists use the word material. In our case it will be useful to differentiate those two expressions: when I say shader I mean Yafaray's specialized algorithms, when I say material I mean the visual result that we create with tweaking the parameters of those algorithms.*

Yafaray (as of 0.1.2) has 5 shaders that allow us to create a lot of different material looks. Those shaders are:

- **shinydiffuse**: Used for a lot of things – from rough diffuse surfaces like wood, transparent & translucent surfaces like fabric or plastic, to highly reflective surfaces like chrome.
- **glossy**: Another very broad shader used to create materials with a specular component like metals, plastic, paint, ceramic, glazing...
- **coated glossy**: like glossy but with a mirror effect that's controlled by IOR value. Coated glossy with IOR=1.0 behaves the same as glossy.
- **glass**: A specialized shader for fast simulation of glass and realistic reflection & refraction of light. Used also for other transparent things like water.
- **rough glass**: An even more specialized shader used for frosted glass

Apart from those there's also a **blend** shader, that blends two others together to create advanced materials with a broad set of properties.

*See also: Yafaray shaders*
3.a Material integration

Yafaray mainly ignores the Blender settings in the material panel (F5). The only tabs that apply to it are those dealing with texture channels: **Texture, Map Input and Map To**. So changing the material diffuse color in Blender won't change it in Yafaray – you must do it in the export script. Yafaray supports multiple materials on an object (per face material).

Firstly you create the material and assign it a name, as would usually. Then you select a Yafaray shader (**material type** drop down) most suitable for your material and manipulate it's parameters to achieve the desired look.

The shader parameters can be modulated (at least some of them) also by textures. You load textures (F6) as would normally in Blender and map them to the desired shader parameters. Not all Yafaray shaders offer the same or all of the possible mappings, like Blender internal does. Which mappings are available to a shader and which shader parameters are modified by them, is written beneath the controls in the **Material** tab of the export script.

See also: **Material modulation & mappable parameters**

Note: The materials in this tutorial won't be very advanced, but will do for an introduction to Yafaray.

3.b Creating our materials

First let's (re)create the material for the table. In this case we will assume that the table is made of varnished wood to get nice reflections of the objects on it. That means that the material must exhibit a mirror effect, so we will use shinydiffuse shader. When you first create a 'blank' material in Blender a default shinydiffuse shader is assigned in Yafaray export script. So, select the table, create the new material and go to the **Material** panel in the export script. You can choose the material by name form the dropdown at the top, or click the **From active object** button, which will show you the (active) material of the active object selected in the viewport. The **Always show active object** toggle will do that automatically.
The table material uses the texture 'wood.C.jpg' mapped to it's diffuse color. The amount of color that the texture adds to the base diffuse color and the type of blending can be controlled in Blender's Map To tab.

We want the table to be made of dark varnished wood, so we lower the diffuse reflection and apply some mirror strength, which controls how much the table will reflect the surrounding environment. We also darken the mirror color, to get a more dark mirror effect. To get a more realistic reflection we enable fresnel, which means that the amount of mirror effect will depend on the angle at which the surface is viewed – there will be more mirror effect at grazing angles, than at perpendicular to the camera. IOR setting controls the fresnel effect strength.

The table uses the Oren-Nayar diffuse reflection model, which is more realistic than Lambertian.
Now we will create the materials for the vases. The exterior of the vase will consist of a rough painted clay look and the border and interior will be more glazed. We will need a shader with specular reflectivity for those, so we will use (coated) glossy.

The exterior has a subtle bump map ('vase.B.jpg') applied to create the impression of roughly painted surface (note: all who use the stable release will have to change the NOR mapping strength to get similar results like on the pictures, as the experimental version uses a different scale). The glossy reflection slider determines how reflective is the surface – the higher, the less diffuse is the surface and the more reflective it gets. The exponent parameter defines how rough is the specular reflection. Highly specular surfaces like smooth plastic need a high exponent, while rough surfaces (as in our case for exterior of the vase) need a low exponent, low being < 100. We will give our exterior material also a tad of mirror effect, so raise the IOR value a little.

For the glazed material of the vase we will also use the coated glossy shader, but with higher glossy reflection, exponent and IOR values. You should compare those two materials (for example apply the glazed one to another vase exterior and re-render), to see the differences. The glossy shader is very versatile.

See also: Glossy shader, Specular color, Specular intensity

The umbrella mesh has also two materials applied. Both use the shinydiffuse shader. The wood material uses the same diffuse color texture as the table, but differently mapped. It uses also a bump map. The only changes to the shinydiffuse default shader were the diffuse color and Oren-Nayar diffuse reflection model.

The paper material uses a color and bump texture ('paper.C.jpg' and 'paper.B.jpg' respectively). Paper is a translucent material (light gets trough), so we will have to set translucency which controls how much light gets trough. We will also give it a tad of transparency.
See also: Shinydiffuse shader, Alpha intensity

Of course all those mappable parameters that we set to a fixed value could be (and should be) controlled by a texture instead.

So, now that we have set all materials, it's time to see how they look like in reality – i.e. under environment and lighting conditions. Disable the clay render in settings and hit Render:
Of course you will seldom get the right look the first time, so some tweaking of materials, lights and re-rendering is needed to get it.

Yafaray supports several blending modes for texture channels, diffuse color stenciling and several procedural textures.

See also: Material modulation, Texture mapping, Texture blending modes, Stenciling, Procedural textures

4. Final notes

* For the explanation of other, non-lighting method related options in the Settings tab, see the Render settings section of the user guide.

* There are generally two anti-aliasing strategies; adaptive AA consists of a lot of passes (say 10 or more) with few samples (1 – 2). An alternative could be to use only one or few passes with a lot of samples (10+). In our example we did not need to use a heavy AA strategy, because there wasn't a lot of noise. You will need to test different AA strategies from scene to scene to see what produces better results.

See also: Anti-aliasing

* When saving the render from the GUI, you can choose to save it with an alpha channel without the background. Enable save with alpha channel in the image menu and save as PNG. Then you can composite your own background in your favorite image editor.

So, I hope this tutorial was somehow a useful intro to Yafaray render engine. If you have any questions ask them in the forums, but before opening new threads, please be sure to exhaust other options like the user guide, FAQ, the tutorials, and the search option, to avoid asking questions that have already been answered.

License note: This pdf can and should be shared, printed, derived from and used freely for personal
or other learning purposes, for presentations or other activities related to the Yafaray project. But its' content cannot be published on personal sites or blogs, instead you should link to the same tutorial present on the Yafaray site

Happy Yafaraying!

~ Matej Mo,