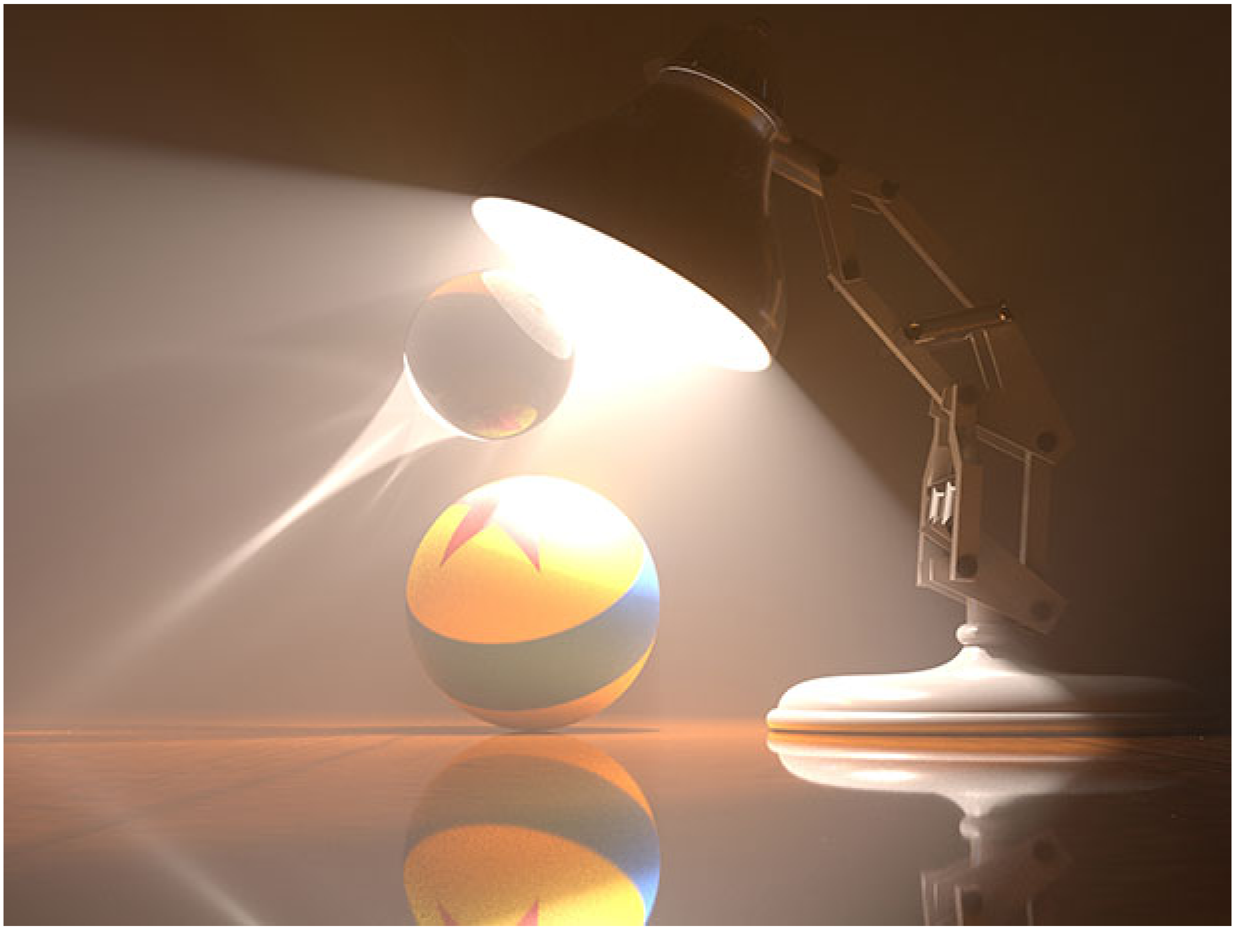
PxrUPBP



PxrUPBP Volume Caustics

PxrUPBP is intentionally not exposed by default. It is included with RenderMan, but you must enable it manually if you want to test it.

- In Maya: alter the exclude list in RenderMan_for_Maya.ini.
- In Katana: add PxrUPBP as an integrator in PrmanGlobalSettings.xml.

This is an **experimental** final-quality integrator in RenderMan. Like PxrVCM it combines bidirectional path tracing with progressive photon mapping (also known as vertex merging). In addition, it stores photon points and beams in volumes. Each of these techniques brings the ability to capture a certain range of light transport paths more efficiently than a pure forward path tracing algorithm. For surfaces, PxrUPBP computes the same results as the PxrVCM Integrator, but PxrUPBP computes better (less noisy) direct and indirect illumination in volumes -- for example crepuscular rays ("God Rays"), volume caustics, and soft indirect illumination. You may notice the controls are nearly identical to the PxrVCM Integrator.

- Photon points and beams improves the convergence of illumination in homogeneous volumes by storing some light paths as volume photon points and beams. This is combined with the techniques below and is the basis for PxrUPBP's superior performance for volume rendering.
- Bidirectional path tracing can converge faster than forward path tracing for interior scenes or scenes with significant indirect illumination. In particular, when forward path tracing can't easily find an indirect path to the dominant light source, bidirectional path tracing can help – especially if the dominant light source is "one bounce" away from illuminating the scene. It also offers the possibility of faster indirect lighting in general because bidirectional path tracing can make multiple connections between a light path and a eye path and reuse shader evaluations. (A forward path tracer has to run a shader at every bounce, and therefore cannot reuse shader evaluations.) Note that this benefit is more pronounced if your pattern evaluations are expensive; Bxdfs without pattern inputs will not benefit from this.
- Progressive photon mapping converges faster for specular-diffuse-specular lighting (reflections or refractions of caustics), especially when dealing with relatively small light sources.

These sampling techniques are combined along with direct lighting (connecting an eye vertex directly to a light source) using multiple importance sampling such that the combined algorithm is substantially more robust and converges faster across a wide range of shots than simply using any one technique by itself.

The PxrUPBP integrator in its default mode of operation enables all of these techniques, but should the need present itself, can also operate as a pure bidirectional path tracer, a pure unidirectional path tracer, or even use pure progressive photon mapping by disabling merging and/or connecting via the mergePaths and connectPaths parameters, and reducing beamsPathsRatio to 0.0%.

For direct illumination, PxrUPBP looks at the values of the numLightSamples and numBxdfSamples parameters in order to determine the number of light vs. Bxdf direct illumination samples. For lowest noise numLightSamples and numBxdfSamples should be set to the same value.

PxrUPBP fully supports multiple scattering of *homogeneous* volumes, including multiple overlapping homogeneous volumes and anisotropic scattering. Bidirectional path tracing, in particular, is well suited for multiple scattering in volumes. PxrUPBP improves performance by supporting a combination of volumetric photon points and beams, giving less noise in direct volume illumination and especially indirect volume illumination such as volume caustics. Inhomogeneous volumes are not yet supported.

There is an important difference to note between this integrator and PxrVCM:

- PxrUPBP introduces photon beams to improve volume rendering. This control is the beamsPathsRatio described below. By default this control is set to 5% of volume photons stored as beams.

There are several important differences to note between this integrator and PxrPathTracer:

- PxrUPBP does not explicitly support the ability to globally turn off caustic light paths (like PxrPathTracer does). However, by default your scene may render the same as if using PxrPathTracer. This is because lights control the tracing of these rays using the **Trace Light Paths** parameter which is off by default. You should selectively turn on this feature to produce the effects you need visually rather than having them all on which may prove costly and be more difficult to art direct. (LPEs can be used to remove caustic paths in PxrUPBP but the calculation is still performed.)
- PxrUPBP is less forgiving for having different numLightSamples than numBxdfSamples than PxrPathTracer. These two parameters should (nearly) always be be set to the same value -- otherwise unnecessary noise can appear.
- Due to the nature of bidirectional path tracing, PxrUPBP does not expose any controls over the indirect ray sample count; all vertices generate at most one indirect ray.
- Rendering a crop window may result in a different quality render than a full frame render. This is because light paths are generated per-pixel and stored in a photon map. When rendering a smaller region, fewer photons are saved as a side-effect. In theory all the light paths could be generated but the impact on performance would begin to erode the benefit of rendering a small region of the final image.

For more information, please consult the relevant papers:

- Jaroslav Kivánek, Iliyan Georgiev, Toshiya Hachisuka, Petr Vevoda, Martin Sik, Derek Nowrouzezahrai, and Wojciech Jarosz. Unifying Points, Beams, and Paths in Volumetric Light Transport Simulation. ACM Transactions on Graphics (Proc. SIGGRAPH), 2014.
- Iliyan Georgiev, Jaroslav Kivánek, Tomáš Davidovi and Philipp Slusallek. Light Transport Simulation with Vertex Connection and Merging. ACM Transactions on Graphics (SIGGRAPH Asia 2012).
- Toshiya Hachisuka, Jacopo Pantaleoni, and Henrik Wann Jensen. A Path Space Extension for Robust Light Transport Simulation. ACM Transactions on Graphics (SIGGRAPH Asia 2012).

Parameters

Parameter	Description
connectPaths	Controls whether "vertex connection" is employed. The power of bidirectional path tracing over forward path tracing is in the better handling of indirect illumination from blocked light sources, leading to faster convergence for interior scenes with significant indirect illumination. As a bonus, the integrator can repeatedly reuse indirect contributions down the full combined path by making multiple connections between the eye and light paths. Set this parameter to either 0 (vertex connection off) or 1 (vertex connection on). By default, vertex connection is enabled. With path connection enabled, but no vertex merging, UPBP acts only as a bidirectional path tracer.
mergePaths	Controls whether "vertex merging" is employed. Vertex merging enables a variant of progressive photon mapping, which converges faster than path tracing for specular-diffuse-specular (caustic) lighting. Enabling vertex merging is generally recommended but does result in additional time and memory overhead for photons, and may be unnecessary in scenes with little specular-diffuse-specular transport. Set this parameter to either 0 (vertex merging off) or 1 (vertex merging on). By default, vertex merging is enabled. Note that photon generation requires the incremental flag be turned on for the Hider; the photons generated in the current iteration are used as the photon map for the next iteration. With vertex merging enabled and vertex connection disabled, UPBP implements a variant of bidirectional progressive photon mapping; this mode is generally not recommended as it does not make direct connections between the eye vertex and the light, and as a consequence usually has slower convergence for direct lighting than all other modes. With both vertex merging and vertex connection disabled, UPBP acts only as a forward, unidirectional path tracer (no light paths are generated).
mergeRadius	Specifies the radius used in vertex merging. It is measured in screen space pixels (not world space). Increasing this radius will blur the influence of a photon over a wider region, which may be helpful in reducing noisy caustics. However, increasing the radius will also slow down merging and increase the initial bias of progressive photon mapping, requiring more iterations to arrive at a bias-free result. The default value of 5.0 significantly reduces noise at the cost of some initial bias which decreases with more samples. Should these bias artifacts be objectionable when using low numbers of samples, decreasing the mergeRadius will reduce these artifacts at the expense of more noise.
timeRadius	Specifies the maximum difference in time, measured as a fraction of the shutter interval, over which photons are merged. By default, the timeRadius is 1.0: eye vertices can be merged with photons emitted at any time. This may lead to inaccurate results (i.e. smearing in time) for caustics from moving objects. Setting the timeRadius to a smaller value tightens the search radius in time for mergeable photons, leading to better results for motion blurred caustics, at the cost of static objects requiring more photons in order to resolve caustics.
maxPathLength	Controls the maximum length of a combined light and eye path, including the connection rays. For example, a value of 4 will permit up to 3 bounces of global illumination. A value of 1 for this parameter will allow direct illumination only, which means that no light paths will be generated (since eye path vertices are always directly connected to the light source). The default value of this parameter is 10.

numLightSamples	Controls the number of light samples for direct illumination per vertex on the eye path. The default is 1. For lowest noise, numLightSamples and numBxdfSamples should be set to the same value.
numBxdfSamples	Controls the number of Bxdf samples for direct illumination per vertex on the eye path. The default is 1. For lowest noise, numLightSamples and numBxdfSamples should be set to the same value.
beamsPathsRatio	Determines what fraction of light through volumes is stored as photon beams in addition to photon points. For sparse volumes more beams reduces noise but increases render times. The default value is 5% or 0.05.
rouletteDepth	Controls the path length at which the integrator begins performing Russian roulette in order to reduce overall path length. The default is 4. Note that Russian roulette is applied separately to both the eye and the light paths.
rouletteThreshold	Controls the path throughput threshold below which to begin performing Russian roulette. The default is 0.2. Increasing the threshold will lead to a reduction in path length, which will lead to an overall speed-up at the expense of higher noise.
clampDepth	If a value for the clampLuminance parameter is specified, then clampDepth controls the ray depth at which to begin clamping based on the per-ray luminance. For example, setting this parameter to 2 and also specifying a value of 4 forclampLuminance will ensure that the luminance of each ray's contribution is no more than 4 for all indirect illumination, without affecting or clamping the direct illumination. The default is 2.
clampLuminance	By default the PxrUPBP integrator clamps the luminance computed by any technique on any vertex to be at most 10.0. However, it is possible to change this behavior by specifying a different value for the clampLuminance parameter. Specifying a relatively low value for the clampLuminance parameter (for example, between 2 and 20) can greatly speed up convergence. In some cases, indirect illumination lights paths may be noticeably dimmer due to clamping; this may be an acceptable trade-off in certain cases. Setting this parameter to a very large number (such as 1e30) will effectively disable all clamping. The default is 10.0.
photonGuiding	Sets the probability of using photon guiding during photon emission. A value of 0.0 means no photon guiding. Values between 0.0 and 1.0 cause a combination of photon guiding and standard photon emission. Finally, a value of 1.0 means that all photons are guided – which may result in biased images. The default value is 0.0.
photonGuidingBBoxMin **photonGuidingBBoxMax**	These two values can be used to explicitly specify the bounding box (in world-space coordinates) that photons should be emitted toward. I this bounding box is not specified, one will be computed automatically as a slightly (loose) bounding box of the directly visible parts of the scene. The default value for photonGuidingBBoxMin is (1e30, 1e30, 1e30) and for photonGuidingBBoxMax is (-1e30, -1e30, -1e30).

Standard AOVs

On top of regular LPE-based AOVs, this integrator outputs a number of standard AOVs typically used by compositors.

Declaration	Contents	Channels
color __Pworld	P in world-space	__Pworld.r : x component __Pworld.g : y component __Pworld.b : z component
color __Nworld	Nn in world-space	__Nworld.r : x component __Nworld.g : y component __Nworld.b : z component
color __depth	Multi-purpose AOV	__depth.r : depth from camera in world-space __depth.g : height in world-space __depth.b : geometric facing ratio : abs(Nn.V)
color __st	Texture coords	__st.x : s __st.y : t __st.z : 0.0
color __Pref	Reference Position primvar (if available)	__Pref.r : x component __Pref.g : y component __Pref.b : z component
color __Nref	Reference Normal primvar (if available)	__Nref.r : x component __Nref.g : y component __Nref.b : z component
color __WPref	Reference World Position primvar (if available)	__WPref.r : x component __WPref.g : y component __WPref.b : z component
color __WNref	Reference World Normal primvar (if available)	__WNref.r : x component __WNref.g : y component __WNref.b : z component