Podium V2 - Customising presets
FIRST DRAFT

Introduction

Podium presets are complicated and have lots of parameters you can set. The benefit is that once you understand what they do, you have a huge amount of control over the render engine and the effects it has on your scene.

This document will explain the basic principles of the approach to help you to understand what the various parameters do, the principles behind them, and the effect they have on the final rendered image. The content is a lot more technical than Podium users are used to, and illustrates the extent to which the render process has been simplified for people who don't want to be bothered with learning the detail and the terminology. All the complexity is hidden by using a series of common configurations that should work equally well for a variety of scene types. It is logical that not all parameters need to be modified for all types of scene, and that if many of the parameters only need to be changed for certain types of image, then a series of presets can be created to deal with almost all situations.

However, the process of creating presets is time-consuming and introduces compromises for certain types of image. Therefore under certain circumstances some people may want to create their own preset configurations. Anyone that successfully creates them is encouraged to share them on the Podium forum where we will create a thread specifically for tposting presets.

The Photon Mapping algorithm

Podium 2's new render engine supports a number of different render algorithms. The main one that will be almost always used is photon mapping with irradiance caching. You can read more about this here and here. A 'photon' is the basic element of light and this approach is based on creating a 'light map' of the scene.

This is based on physically accurate principles, but relies on a few tricks and a little common sense to get a very good (though not 100% perfect) simulation but in significantly less time. This approach is used by a number of render engines, and it is what is referred to as a 'biased' solution because it is based on certain assumptions about how light will behave. The photon mapping implementation differs slightly from one renderer to another in terms of refinements relating to what assumptions are made, and how the calculations are carried out.

Rendering involves 2 stages. Initially an approximate preview pass is made, which shows the user a very noisy view of their scene complete with textures, shadows and reflections. This is followed by the photon mapping stage which simply deals with lighting and calculates an approximation of the lighting levels for the scene. It is based on dividing a scene into a series of small cells with a similar lighting level. Logically, some parts of a scene have more or less the same lighting levels as others, particularly neighbouring areas, so if a scene is divided into a smaller series of zones which are all treated the same, and the final result is blended together, you should end up with a sufficiently accurate image for photorealism.

Obviously the smaller the cell size, the greater the number of cells will be created and the more accurate (and slower) the final render will be.

If every individual pixel or photon was calculated individually, render time would be increase massively. This is what 'unbiased' render engines do. The results are as accurate as possible because the scene is a physical simulation of what happens in reality. There is no need to tweak the render parameters because everything works on the laws of physics and in theory you get a perfect simulation.

The trick lies in judging how many cells to create, and how to divide the scene. Areas of high contrast need a greater number of smaller samples, whereas say large areas of blank wall can have fewer, larger cells.

As mentioned earlier, Podium's engine has a number of differences which enable it to create extremely high quality images much more quickly than most other render engines.

Following the photon mapping stage, our engine makes one or more prerender passes which refine the calculations before the final render phase. Although there are additional stages, the time taken for these is to an extent offset by reducing the time taken for the final stage.

This is called 'Final Gathering' and it refines the solution calculated by the preceding stages. It is almost like a separate calculation, but is based on a refinement of an existing data set.

Configuring the render parameters is therefore based on selecting sensible values for the photon mapping, prerender and final gathering phases.

This document deals with each render phase as a separate section.

General Parameters

__blrthrsh - blur threshold. ,0.0001

__*Octdepth* - octree depth. This has the greatest effect with large scenes and sets the size of the data structure to hold scene information. The higher the value the more system RAM is used. The higher the value, the faster the render if the system has enough memory installed. Systems with less RAM can reduce this value to allow background tasks to run effectively without the machine freezing. Effective values range from 50 to 350. For systems with 2Gb of RAM the default value of 150 is sufficient. With 4Gb on a Mac, this can be increased to 250 without problem. For very powerful machines with 8Gb or more values of 350 should be no problem. No testing has been done with values greater than this.

__*material_reflection_blur_default* - the default % blur factor for all reflective materials. To simplify the interface reflection blur is configured globally instead of per material. The default is 50% which translates as a parameter of 0.5

__*material_refraction_blur_default* - the default % blur factor for all refractive materials. This is the same as the above parameter, but for refraction. Blurred refraction is 'true' translucency.

__lem_power_multiple - Global multiplier for LEM power. Increasing this value makes LEMs more powerful generally, default 10

__turbidity - turbidity is the haziness of the sky based on the sun reflecting off particles in the lower atmosphere. It is what makes the horizon a different colour. The effect is most noticeable towards sunrise and sunset, when the sun is low in the sky. Lower values represent clearer skies, higher values represent hazier skies. The effective range is from 1.5 to around 4. The default value is 2, but care should be taken when adjusting this parameter, small variations e.g. 1.9 or 2.1 can have a noticeable effect.

__sky_exposure - the overall brightness of the sky. This determines the global illumination strength and if set too high will cause details and textures in full sun to be washed out. The effective range is from 0.8 to 3. The higher end of the spectrum should only be used for interiors to get more light through windows. The default value for exteriors is 2

Photon Mapping

__gi_res - global illumination resolution. This acts as an overall scaling factor for all the other parameters. Reducing it increases the fineness of detail but increases render time, default 1m.

__usecaustics - flag to enable caustics if a render mode supporting it is selected, default 0

__pmphotons - number of photons. Positive numbers are the photons fired, negative numbers set the number of photons received by the scene, default -200000

__pmN - photon mapping N - number of photons in a sample, the greater the number, the finer the detail. This works with the number of photons to define the number of cells, default 600.

__oversample,0.5

__ppdist - precached photon distance. This determines the size of the cells in each scene. It is a percentage of the GI resolution. The larger the scene, the less accurate the GI will be for any given value. Default value is 50% of GI resolution, which is translates to a variable value of 0.5 and a distance of 500mm/50cm

__pblur - photon blur. This is a photon blur factor for GI cells, similar to pmN. The default value is 2

__cphotons - number of caustics photons. Similar to *__pmphotons*, default -100000

__cN - equivalent to *__pmN* for caustics photons

__lightbounce - configures the number of light bounces to calculate in the scene. The more bounces, the better results are obtained when rendering scenes with multiple levels of transparency and reflectivity. Our render engine processes multiple light bounces extremely quickly, so increasing this figure doesn't have an enormous speed penalty. A minimum value is 10, but 20 can be used. Values above this are unlikely to have any benefit. Modifying this is not recommended.

Final Gathering
This is the final stage of the actual render process. The stages after this (tonemapping, pixel filtering and antialiasing) affect the final rendered image.

__FGRmin - final gathering minimum rays. The minimum number of rays used by final gathering for each sampled point. Use higher values for more quality at cost of speed, default 50

__FGRmax - final gathering maximum rays. The maximum number of rays used by final gathering. If set too low, blotches will occur. This parameter determines quality but has a direct effect on render speed. Try to use the lowest value that gives the best result. The default for exteriors is 200. Use more for interiors, but no more than 2000 for very highest quality

__FGth - final gathering threshold. This is the difference between pixel values allowed before more rays are fired, default 0.0001

__FGSPtol - final gathering Spatial tolerance, this is the smallest distance between 2 sampled points. It determines shadow accuracy. Values should change depending on scene size. Exteriors can use much higher values than interiors. For exteriors, the default is 0.3, For interiors values of 0.05 to 0.1 are better.

__FGangTol - final gathering angle tolerance. This determines sampling on curved surfaces and the angle change before new samples are added, default 30

__FGblur - final gathering blur. Sets the extent to which neighbouring pixels in sampled areas are blurred. It can reduce splotches with a low number of FG samples to improve speed, but can wipe out fine detail. Use values from 0 to 5, default 4

__FGdistmin - final gathering minimum distance. This is the length of the ray from the sampling point. Ray distances shorter than this are treated as this distance, default value 0.4. This is not normally modified

__FGdistmax - final gathering maximum distance. Similar to above, if the distance between any ray to the sampling point is greater than this, it is treated as the same distance, default value 10. As above, this parameter is is not normally modified

__BD - brightness/density, adds more samples in high contrast areas. Preferable to use prerender steps to refine detail. Range from 0 to 1, default 0

__prerendlevel - prerender level. Percentage of pixels to be prerendered. The percentage of the scene prerendered increases render time for this phase but should reduce the render time for the final render pass. Use 0.5 to 1 for best results = 50-100%, default 1

__prerenddiff - prerender difference. This is the difference between pixel values allowed before another pass is required. It can reduce blotchiness in the scene. Higher values reduce sensitivity and increase speed, default 0.05

__prerendpass - prerender passes. The number of passes for the prerender stage. Prerendering increases accuracy, but the more passes are made, the longer the render

time. To an extent, the increase in time is compensated for by a faster final render pass. Default value is 1

__ptCornDist - path tracing corner distance. Photon mapping can create problems in corners, so the slower, but more accurate path tracing algorithm can be used here instead. This variable sets the rendering distance from the corner, default 0.5

__ptCornPath - path tracing corner paths. The number of passes for calculating corners. When set to greater than zero, a path tracing algorithm is used for the corners. This can reduce blotches in corners, default 1

Antialiasing

This is effectively smoothing out jagged edges in a scene. In some cases you might want to avoid this stage altogether, and render a larger image than you need, resizing it in an image editor afterwards. For scenes where the AA pass takes a very long time, you might want to consider this approach as an alternative, as the algorithms used for resizing images tend to smooth out edges. There are 2 sets of parameters to consider, edge sampling and antialising method. The default type is grid or rotated grid, which analyses pixels based on an *n* x *n* grid. Both grid size and edge smoothing parameters can be configured.

__undersample_edge - edge undersample. This can reduce render time by reducing the resolution of the first pass. A factor of 0.5 will reduce the resolution by 50%. The default value is 1 which effectively disables the feature

__undersample_thrsh - undersample threshold. If the difference between 2 pixel values exceeds this, the pixels are sampled at full resolution, default 0.0001

__undersample_px - undersample pixels. The number of pixels to undersample by, default value is 0, but values of 1-4 can be effective.

__EdgeAbs - edge absolute. Sets the absolute difference between two pixels. If the difference is greater, pixels are treated as being on an edge and are not smoothed, default 0.1

__EdgeRel - edge relative. Sets the relative difference between pixels for smoothing. If absolute differences between pixel values is small, edges can be missed by edge absolute. Edge relative is more precise, default 0.1

__EdgeNorm - edge normal. This sets smoothing for geometry edges and normal/ bumpmap edges, default 0.1

__EdgeZ - edge z difference. Detects differences in depth based on distance to camera. This will pick up shadows from small recesses even with dark colours, default 0.1

__EdgeUp - edge upsampling. This treats images as if they are higher resolution in terms of pixel sizes for edge width so tiny details will not be missed for smoothing, default 0

__EdgeThick - edge thickness. This determines the thickness of the edges around the area for smoothing, default 1

__EdgeOver - edge overburn. This setting determines whether edges on overburned areas should be smoothed, default 1

__aa_gridsize - AA grid size. The size of the antialising pixel grid, default 3

Pixel filtering & Tonemapping

These are 2 distinctly different features, the only thing they have in common is that they both process the final rendered image. They are grouped together only for convenience.

Pixel filters are just like Photoshop filters. There are a number of different options and they can make your image look distinctly different. You may have seen edge sharpening and softening filters in image editors, these perform the same sort of operations. The 2 most commonly-used ones for Podium are Lanczos, which makes images look sharp and are generally better for lower resolutions, and the Mitchell filter which sharpens and softens the image. Compared to the Lanczos filter, it makes images look distinctly softer and produces a more naturalistic photographic feel. Some people feel that Mitchell filters produce images which are slightly 'fuzzy'. It is best used on the largest resolution images where an overly-sharp image looks too much like CGI and a little unnatural. A Box filter can also be used and gives quite a sharp image. There are more pixel filter types which won't be documented here, but they are less important.

__pxFiltRad - pixel filter radius. Radius for processing edges, mainly used for the Lanczos and Box filters, default 0.5

Tone mapping is a way of processing standard definition computer images to simulate the appearance of high dynamic range images. In life, the eye can capture a much wider range of tonal values than can be easily simulated on a computer monitor. As a result, the darkest and brightest areas are shadowed and washed out respectively. Using tone mapping enables a wider range of tonal values to be simulated so that detail is retained.

This option is only required if .png format images are ceated. If you choose .hdr output it is better to post-process your image to adjust brightness, contrast, saturation etc.

There are two basic tone mapping modes, gamma and exponential. Gamma mainly affects scene brightness and contrast (just like adjusting brightness and contrast in an image editor) whereas Exponential tends to create very contrasty colour-saturated images.

These two are best used by blending them together.

There are 3 lines in the presets which define this:-

tonemapdefine t1,gamma_exposure,1,1 - defines a tonemapper configuration called t1, with parameter 1 and exposure 1

tonemapdefine t2,exp_exposure,2.2,1 - defines a tonemapper configuration called t2, with parameter 2.2 and exposure 1

tonemapper blend,0.4,t1,t2 - blends tonemappers assigning 40% to t1, which means that t2 contributes 60% of the total.

Light quality

__LLthrsh - Linear light threshold. This defines the sampling for linear lights. Lower values increase accuracy but also render time, default 0.002

__LLRmin - Linear light minimum recursions. This defines the minimum number of recursions for linear light calculations. Higher values produce less noise and higher quality but increase render time. This is not normally modified, default 1

__LLRmax - Linear light maximum recursions. As above, this defines the maximum number of recursions. This is not normally modified, default 4

__ALthrsh - Area light threshold. Similar to *__LLthrsh*, this defines the sampling for area lights. Paramter values are otherwise identical

__ALRmin - Area light minimum recursions. The same as *__LLRmin*, but for Area lights with identical parameter values.

__ALRmax - Area light maximum recursions. The same as *__LLRmax*, but for Area lights with identical parameter values.

__AreaDouble - Area light doublesidedness. A flag to configure whether area lights are double sided. Functions as a switch with a value as 1 to enable this, default 0

__LumThrsh - Luminosity threshold. ,0.01

__Rmin - Minimum rays. ,10

__Rmax - Maximum rays. ,60

__blur_accuracy - This determines the accuracy of blurred reflection & refraction. Lower values increase speed, higher values increase accuracy. The range for values is 0.5 to 1.5, default 1.

__blrthrsh - Blur threshold. This is the difference between pixel values before the render engine , default 0.0001

__blur_rays_min - Minimum blur rays. The minimum number of rays to be fired to define blur. The more rays are fired, the higher the quality and the slower the render. Minimum value is 10, a reasonable default is 400

__blur_rays_max - Maximum number of blur rays. The more rays are fired, the higher the quality and the slower the render. Minimum value is 100, a reasonable default is 400

__blur_noise - Configures the amount of noise. Smaller values are more accurate but take longer to render. Default value 0.001.