Lighting technology of "The Last Of Us"

Michal Iwanicki*
Naughty Dog, Inc.

1 Introduction

"The Last Of Us" is an upcoming action-adventure, survival-horror game from Naughty Dog for the Playstation 3 console. Since lighting plays an important role in the game, we decided to use precomputed lightmaps to deliver the highest possible quality lighting at interactive framerates. Even though they are commonly used, lightmaps still suffer from some distracting artifacts - like lighting discontinuities resulting from *uv* layout, lack of specular highlights and absence of shadows from dynamic objects. We utilize a simple, directional representation of incoming lighting to solve these problems. The methods are simple and effective in terms of both production time and real-time performance and produce a result of quality unseen in games before.

2 Directional lightmaps

Our preprocessing pipeline starts by generating spherical harmonics (SH) lightmaps [Good and Taylor 2005]. From this representation a dominant light direction is extracted as described by [Sloan 2008]. Since this direction tends to change rapidly on shadow boundaries, we apply a low-pass spatial filter to this data, which eliminates some artifacts in later stages. In the next step we use the extracted direction, together with original SH lightmap, to convert incoming lighting into a pair of lights, one ambient and one directional, such that the error in the reconstruction is minimized in a least-squares sense. This simplified representation is used at runtime. Unlike traditional lightmaps, which only store the diffuse response of the underlying material, ours allow us to utilize normal mapping to show local surface details as well as to provide view-dependent highlights from the directional portion of the extracted lighting.

3 Seam stitching

Lightmaps created this way provide directional information about the incoming light, light, however they can suffer from discontinuity artifacts, similar to traditional lightmaps. Since lightmaps require unique *uv* mapping, it may happen that the mesh needs to be split into parts that are disjoint in *uv* space. This can cause visible lighting discontinuities at runtime.

We solve this by "seam stitching", which we execute in a preprocessing stage. For each edge that is split in *uv* space, we create a number of stitching points placed evenly along it. For each one of them we compute an error value as a difference between the lighting values sampled from the lightmap on both sides of the split edge. We assemble the equations describing error for all the stitching points into a single system, together with a set of equations that ensure that corrected lighting values stay in proximity of the originally calculated ones. User-provided scaling factor controls the ratio between stitching strength and preservation of the orignal values. The entire system is minimized in the least square sense and lighting values stored in the lightmap are updated appropriately. The method can be used with existing parametrizations and doesn't incur any additional cost at runtime.

*e-mail:michal_iwanicki@naughtydog.com

ACM 978-1-4503-2261-4/13/07

Figure 1: Character casting soft shadow onto a lightmapped environment.

4 Soft shadows

Lightmaps contain information about the lighting reflected only from the static parts of the scene and any dynamic objects introduced at runtime don't affect them. Since recomputing the full lighting solution is too costly to be performed every frame, we use the directional information in the lightmap to simulate occlusion at runtime. We approximate the occluder's shape with a set of spheres. The occlusion of ambient component is computed analytically, as the percentage of the hemisphere occluded by a sphere. For the directional component we check for the intersection of a cone originating in each visible world position and the occluding sphere. This can be efficiently done at runtime by rendering the dominant light direction to an off-screen buffer and processing it in a deferred fashion. The cone angle is set by the artist and controls the size of the penumbra. To reduce the amount of computations required, we also limit the influence of each occluder to a certain distance. Occlusion from individual occluders is accumulated, and while in theory this may result in double occlusion, it is not visible in practice.

For rigid objects that would require too many spheres to approximate, we precompute the occlusion in the space around them using Monte Carlo methods. The occlusion function is approximated with a single cone representing the occluded direction and stored in a 3d texture. For those objects we compute the intersection of the precomputed occlusion direction and the cone oriented along dominant light direction.

Shadows created that way don't suffer from the artifacts common to the screen-space methods. They have a soft and natural look that hasn't been shown in games before.

References

GOOD, O., AND TAYLOR, Z. 2005. Optimized photon tracing using spherical harmonic light maps. In *ACM SIGGRAPH 2005 Sketches*, ACM, New York, NY, USA, SIGGRAPH '05.

SLOAN, P.-P. 2008. Stupid spherical harmonics (sh) tricks. In *Game Developers Conference*.