# Realistic Lighting of Architectural Projects Based on IBL Method

Krzysztof Wrobel*

Computer Graphics Group
Szczecin University of Technology
Szczecin / Poland

## Abstract

The aim of this paper is to present new approach to realistic rendering of architectural objects, which simplifies scene lighting preparation process, and shortens rendering time. We present how to apply Image Based Lighting in visualization of architectural projects, and how to implement it so that rendering takes at most few seconds, even for complex 3D models. Our results show that this approach can be used in practice successfully, and that there are many ways to refine the method, and obtain images with even better quality.

**Keywords:** architectural visualization, CAD/CAM, image based lighting, high dynamic range imaging, image synthesis

## 1 Introduction

Currently dedicated CAD systems are used to develop architectural project. During the design process, architect draws the shape of a building as a 3D model. Such model has to be presented to the investor for final approval. But most CAD systems, although very efficient in design, are not effective in producing realistic visualizations of 3D models.

We decided to develop and implement software that would import 3D architectural project from a CAD environment, and automatically produce realistic renderings, at minimum effort from user. Visualization process consists of: conversion of material and geometry form CAD project, lighting configuration, rendering, tone mapping and visualization.

One of the most difficult and time consuming parts of the process of creating realistic 3D visualizations is configuring scene lighting. Average architect finds it difficult to setup position, direction, luminance and color of light sources in a scene. Even experienced architects are not will to spend many hours on this task. Therefore we decided to replace this process with something much more user friendly - Image Based Lighting (IBL) technology [1]. In our solution user needs only to choose High Dynamic Range (HDR) image that he wants to use for scene lighting - and light sources will be setup in the scene automatically. We use IBL method which finds light sources in HDR image, then maps their position onto a sphere surrounding 3D model, and assignees appropriate properties to all light sources in the scene.

There are many programs dedicated to architecture design, one of the most popular is Autodesk Architecture (the latest version: Autodesk Architecture 2008). This is complex and expensive system that doesn't provide good enough visualization tools. Autodesk Architecture rendering engine produces poor quality images and visualization computations are rather slow. Because of this, to create professional and realistic visualizations, additional programs are used by professional architects. There are a few such programs in a market: Autodesk 3ds Max, Blender, Maya, Autodesk Viz, etc. All of them are capable of producing realistic high quality images. They use similar global illumination rendering algorithms: photon mapping or radiosity methods supported by ray tracing and scanline algorithms. The main drawback of these methods is long computation time. Usually many test images have to be created in architectural visualization before achieving desired effect. This is why shortening rendering time is so crucial.

This drawback was recognized and efforts to find solution were undertaken. In 1997 Debevec and Malik [2] described an algorithm of recovering high dynamic range Radiance maps from photographs. In the following paper [1] the Image Based Lighting (IBL) method was presented and both theoretical and practical aspects of realistic rendering based on IBL were discussed. In this paper we present adaptation of this method to visualization of architectural projects.

In section 2, detailed scheme of our algorithm is presented and each step of the algorithm is explained. In section 3 we reveal implementation details and present achieved results. Both quality of acquired images and time needed to render them are discussed. In the final section we conclude the paper and propose future work.

## 2 Realistic rendering of architectural objects

In this section we present rendering method that allows obtaining high quality images of artificial architectural objects. The method is based on the IBL algorithm. Their

*krzysztof.wrobel@gmail.com

results are comparable to global illumination methods but computational costs are significantly smaller. It uses HDR images for realistic illumination of rendering scene, and because of that, rendering process is enriched and simplified. Only by changing light probe, we can light up objects with morning like lighting, evening light, lightening captured in a sunny or cloudy day, or even with a light sample obtained directly at the location of future building.

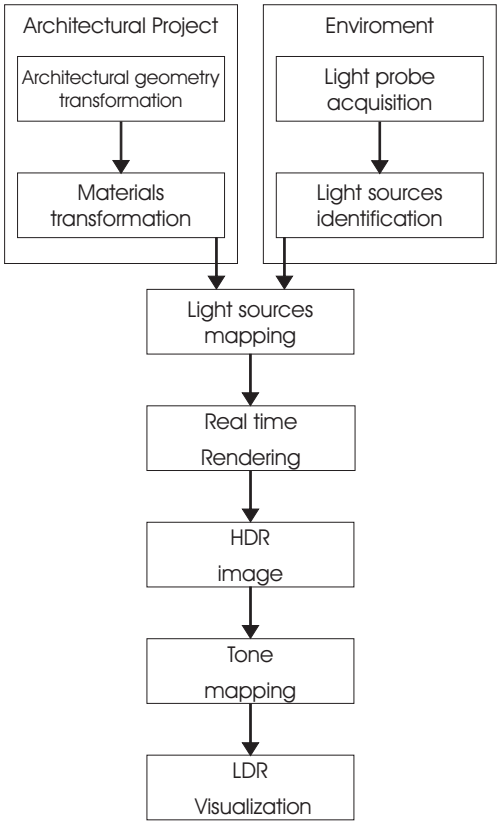


Figure 1: Algorithm generating realistic images of architectural objects. Starting from geometry transformation through light computations, real time rendering, post processing and visualization of generated picture

In Figure 1 a proposed algorithm for realistic rendering of architectural objects is presented. We assume that the starting point is when we have a 3D model of an object. Most architecture dedicated programs use their own file type to store data. Therefore we need to import model data, such as: meshes, materials, texture and camera coordinates from an exchangeable format. In our project we use 3ds format, one of the most popular 3D geometry formats in the Internet, supported by various modelers.

Since color and pattern of an object is very important in visualization of architectural projects, we have to import it along with geometry. Textures are simply 2D images, but can be stored in various formats. Most popular ones are bmp, jpg, gif, png, tga. To be able to handle many formats we use SDL_Image library.

In our approach we do not use whole light probe in rendering stage. We use only points identified as light sources. The problem of identifying light sources is not trivial and different approaches to this problem can be found in literature. Ostromoukhov et al. [12] proposed hierarchical importance sampling algorithm, based on the Penrose tiling. Kollig and Keller [4] based their algorithm on Lloyd's relaxation method. Agarwal et al. [8] proposed an approach that combines elements of importance and stratified sampling. Last but not least paper describing approach based on Monte Carlo algorithm was presented on CESCG 2007 by Maciej Laskowski [5]

Identified light sources have to surround objects being lighted. Therefore light sources can be mapped onto a cube, sphere or hemisphere that surrounds 3D geometry. Mapping lights onto a sphere can be easily obtained with use of spherical coordinate system. First step is to convert position of light in 2D HDR to relevant position in spherical coordinate system (from 2D Cartesian space to spherical coordinates), and then convert new values to final 3D space coordinates. In this space geometry of architectural object must be also defined.

Preparations described earlier provided all the data necessary for rendering. Approach to rendering described in this paper is significantly different form what was presented in 1998 by Debevec [1]. Since 2002 when fourth generation of GPUs was introduced, it is possible to run custom build pixel and fragment shader programs on graphic cards. Taking advantage of this, to achieve best quality at affordable time cost Blinn-Phong shading model was implemented, so that it can run directly on GPU, and is computed for each fragment of a rendered image. Unlike global illumination algorithms, IBL can be rendered in real time by modern GPUs even with hundreds of light sources lighting a scene Last part of rendering process is applying TMO (Tone Mapping Operator). Rendered HDR images contain much higher luminance than the usual LCD can display. Therefore it is required to compress that luminance to the range supported by computer LDR (Low Dynamic Range) monitor. Good effects are achievable with use of gamma compression and clamping (please see [3] for details). Presented algorithm is very flexible. It could be easily extended - especially in the last part, where rendered HDR image could be refined, and special effects could be added to create even more realistic visualization. It is also possible to skip some steps, like acquiring Light Probe, because we can use images that are available on the Internet.

## 3 Implementation and results

In this section we present the software for realistic rendering of architectural projects. The implementation scheme, environment and additional libraries used in the software are described. We also discuss testing environment and achieved results.

In Figure 2 main modules of software architecture are

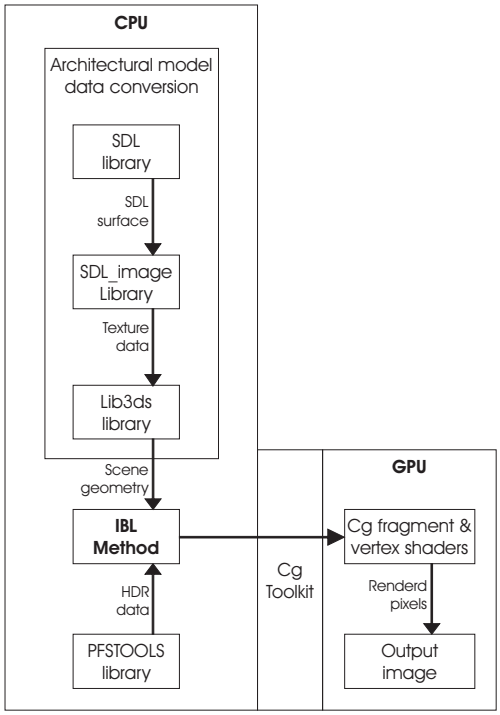


Figure 2: Architecture of the software for realistic rendering of architectural projects. Separate CPU and GPU processing pipelines are presented. External libraries, except for Cg Toolkit, are available under GNU Lesser General Public License

presented. The module responsible for loading architectural projects consists of three separate libraries. *Lib3ds* library [6] (version 1.3) is used to access 3D model data (e.g. meshes, materials and texture coordinates) stored in the .3ds file format. Since textures can be stored in various formats, the *SDL_image* library [10] (version 1.2) is used to decode this formats and write pixel values to the *SDL Surface* structures. The *SDL Surface* class is implemented in the *SDL* library [9] (version 1.2).

Separate part of the software handles HDR image operations. The *PFSTools* library [7] (version 1.6.2) methods and classes are used to load, store, and manipulate Light Probe images. HDR image used for scene lighting is loaded into the program in RGB format. To obtain luminance values of each pixel, we convert image to XYZ format.

IBL method implemented for this presentation, finds 49 pixels with biggest Y component, and treats them as light sources. The number of light sources could be increased to obtain even more accurate renderings. Position of all light sources is mapped from image space to world space coordinates, and then used in color calculation process.

Color calculations are made by fragment shaders in GPU in following order. Firstly material properties are acquired. If fragment being processed has a texture assigned to it, texture image has to be sampled. Then lighting equation is solved. Ambient component is computed once for each fragment, whereas diffuse and specular components are calculated accordingly to number of light sources.

Last part of the process is to apply tone mapping operator. Because tone mapping calculations are done in GPU for each fragment separately and usage of local tone mapping would be tricky, we implemented global tone mapping operator. We use gamma operator with gamma set to 1,8.

Vertex shaders are used for model-view-projection transformations.

The software is written in C++ under Microsoft Visual Studio 2005 environment. We use Cg Toolkit (version 1.5), provided by NVIDIA Corporation, to program GPU.

Figure 3: 3D Architectural test model (downloaded from [11]), rendered in Autodesk 3ds Max(most important settings used: scanline renderer, radiosity, advanced shadows, antialiasing, bump maps). We use this rendering as a reference for further comparisons.

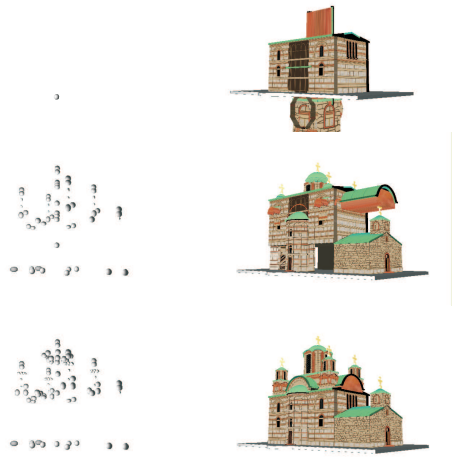

Figure 4: Sequence of transformations necessary to place sub-objects in right object space positions. Images on the left side show view of the model where actual sub-object geometry was replaced with spheres. Images on the right side show actual object geometry.

In the Figure 3 the test 3D model is presented. Like most of architectural models, it is very complex, consists of 113 hierarchically arranged sub-objects (Figure 4), each built with over 30 000 faces and 20 000 vertices. It comes along with 18 high resolution diffuse textures and 18 bump maps.

Figure 5: Three example images rendered by our IBL algorithm (top) and exemplary Autodesk 3ds Max rendering - most important settings used: scanline renderer, radiosity, advanced shadows, antialiasing, bump maps(bottom). The different light probe was used for each IBL rendering

In Figure 5 images of rendered monastery are presented, together with HDR images used for the scene lighting. While selecting light probes, our main concern was not what does light probe actually present, but rather what will be the tint and brightness of light sources selected from it. Also because we wanted to present at least few samples, and the light probe doesn't affect the IBL algorithm itself, we decided to use already existing light probes, rather than creating our own ones. This approach let us obtain significant difference in rendered images. The tint of the building changes together with a light probe that is lighting the scene.

The quality of achieved images is higher than images obtained by OpenGL rendering (see Figure 4 for an example), but Autodesk 3ds Max engine generates higher quality images 3. There are still some optical phenomenon's (like Bump Mapping and shadow rendering) that are not implemented in our software. They have great influence on image quality and further development of our engine should even surpass Autodesk 3ds Max renderings.

The hardware used to render all images presented in this paper consists of:

- Intel Pentium M 760 (2 GHz) 533FSB 2MB L2 Cache,
- 1 GB Ram, PC 4200 DDR2-533,
- NVIDIA GeForce Go 6600 256MB GPU.

| Rendering method | Time to render one frame | Decrease of rendering time in comparison with the IBL method |
|---|---|---|
| OpenGL | 0.5 second | 4x |
| IBL | 2 seconds | - |
| Autodesk 3ds Max | 7 minutes 59 seconds | 240x |

Table 1: Rendering time of the test scene.

Time necessary to render images is shown in table 1. As we can see hardware accelerated IBL is slightly slower than OpenGL rendering. But our method is 240 times faster than Autodesk 3ds Max rendering engine. In many application this speed-up can compensate the drawbacks in image quality between IBL and Autodes 3ds Max methods.

## 4 Conclusions and Future Work

In order to achieve fast generation of realistic renderings of architectural objects, we have presented and implemented a framework that uses IBL method to light up the scene, and then uses GPU to perform fast and accurate color calculations. Presented results show that our approach gives results comparable to Autodesk 3ds Max renderings. Our method generates images of architectural objects, at minimum effort from user. Conversion of 3D model designed in CAD system, is in most cases fully automatic, even for complicated models that consist of many sub models, and use many materials or textures. For our test case result was generated 240 times faster, than in software dedicated to create photo realistic images. Our renderer generates images of almost equal quality in comparison to Autodesk 3ds Max Radiosity renderer. To achieve higher quality images in comparison with Autodesk 3ds Max, other optical phenomenon's should be implemented (e.g. soft shadows, bump mapping, environment mapping). Including additional objects, and a background accompanying the main model, would also contribute to the overall impression. Last but not least, more effective tone mapping operator, (e.g. local version of photographic operator) could be implemented.

## 5 Acknowledgments

The research work, which results are presented in this paper, was sponsored by Polish Ministry of Science and Higher Education (years 2007-2008).

## References


[1] Paul E. Debevec. Rendering with natural light. In *SIGGRAPH '98: ACM SIGGRAPH 98 Electronic art and animation catalog*, page 166, New York, NY, USA, 1998. ACM.

[2] Paul E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 369–378, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co.

[3] Greg Ward Paul E. Debevec E. Reinhard, Sumanta Pattanaink. *High Dynamic Range Imaging, acquisition, display, and Image-Based Lighting*. Elsevier, 2006.

[4] K. Kollig and A. Keller. Efficient illumination by high dynamic range images. In *Europgraphics Symposium on Rendering: 14th Europgraphics Workshop on Rendering*, page 4551, 2003.

[5] Maciej Laskowski. Detection of light sources in digital photographs. In *11th Central European Seminar on Computer Graphics*, 2007.

[6] Lib3ds. Available from `http://lib3ds.sourceforge.net/`.

[7] PFSTools. Available from `{http://www.mpi-inf.mpg.de/resources/pfstools/}`.

[8] S. Belongie S. Agarwal, R. Ramamoorthi and H.W. Jensen. Structured importance sampling of environment maps. In *ACM Transactions on Graphics*, page 605612, 2003.

[9] SDL. Available from `http://www.libsdl.org/projects/SDL_image/`.

[10] SDLimage. Available from `http://www.libsdl.org/projects/SDL_image/`.

[11] TurboSquid. Available from `http://www.turbosquid.com`.

[12] C. Donohue V. Ostromoukhov and P.-M. Jodoin. Fast hierarchical importance sampling with blue noise properties. In *ACM Transactions on Graphics*, pages 488–495, 2004.