An Integrated Scalable Lighting Simulation Tool

Yi Chun Huang

Submitted to the School of Architecture
of Carnegie Mellon University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

School of Architecture
Building Performance and Diagnostics
Carnegie Mellon University

Advisory Committee

Professor Khee Poh Lam (Chair)
School of Architecture
Carnegie Mellon University

Professor Ömer Akin
School of Architecture
Carnegie Mellon University

Associate Professor Doug James
Department of Computer Science
Cornell University

July 2011

I hereby declare that I am the author of this dissertation

I authorize Carnegie Mellon University to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Carnegie Mellon University to reproduce this dissertation by photocopy or other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

____
Yi Chun Huang

Carnegie Mellon University

College of Fine Arts
School of Architecture

Dissertation

Submitted in Partial Fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Title: **An Integrated Scalable Lighting Simulation Tool**
Presented By: **Yi Chun Huang**

Accepted By:

Dan J. Martin Dean Date

Stephen R. Lee Head Date

Khee Poh Lam Principal Advisor Date

Carnegie Mellon University

College of Fine Arts
School of Architecture

Dissertation

Submitted in Partial Fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Title: **An Integrated Scalable Lighting Simulation Tool**
Presented By: **Yi Chun Huang**

Accepted By Advisory Committee:

Khee Poh Lam (CMU)	Principal Advisor	Date
Ömer Akin (CMU)	Advisor	Date
Doug James (Cornell)	Advisor	Date

Abstract

Lighting simulation contribute readily to the synthesis of high performance lighting designs. Unfortunately there exist several issues impeding the pervasive use of lighting simulation, including:

1. Most of the time in preparing lighting simulations is spent towards the input of existing but non-interoperable information between different tools.
2. Lighting simulation tools do not complement integrated building design processes where the design solution is progressively developed in multiple disciplines concurrently; lighting simulation tools require design information (attributes) that may not yet be defined, and is non-interoperable with other tools.
3. Disparate tools with vastly different technical approaches available for different stages of the building design process do not allow consistent or meaningful performance comparisons between design versions, and similarly makes design performance progress tracking between design versions difficult.
4. Lighting simulation tools provide radiance and irradiance values as simulation results, and much time and manual effort is required to process these results into *operative information*, information that is directly applicable in making design decisions.
5. Lighting simulation tools employ outdated rendering[1] techniques that are inadequate in evaluating highly-reflected irradiance[2], a typical feature in high performance building designs.

[1] *Rendering* commonly refers to the process of generating computer images from computer models of three-dimensional objects. In lighting simulation, the objective is to compute physically-accurate radiance and irradiance values (lux and candelas per square meter) within an architecture scene described by some computer model, and these values are then typically presented in the form of two-dimensional bitmap images.

Note that renderings can be either photo-realistic or physically-accurate. While the two are not mutually exclusive, most rendering features found in architecture software applications are focused on the former. *Rendering*, as used in the context of lighting simulation, and the rest of this research, refer to algorithms that produce physically-accurate radiance and irradiance values.

[2] See discussion of contemporary tools (Chapter 1.2.5) and implemented rendering techniques (Chapters 4.1 and 4.2). Highly-reflected irradiance occurs with the use of diffuse lighting strategies, and light re-directing devices such as light- wells, tubes, and shelves, which are common in contemporary high performance, or green buildings.

While there remain other shortcomings in lighting simulation tools as identified by contemporary research[3], the issues above relate closely to the overall effort and time-cost factors attributed to using simulation tools, which has been consistently identified[4] as obstacles towards using simulation tools. This research seeks to reduce the effort and time-cost required to conduct lighting simulation by addressing the issues above. Case studies of actual design scenarios are used to establish quantitatively the effort and time costs baselines for comparison.

The effort and time reduction goal is structured as the following objectives in a new lighting design support tool:

1. Reduce the time and effort to set up and conduct lighting simulation by using interoperable information (building information models) from design modeling tools.
2. Complement integrated design processes by supporting design models of varying completeness[5], in a format that is interoperable with tools from other disciplines in the design team. All information, including assumptions, must be consistent across all disciplines.
3. Provide ability to use performance metrics and consistent technical approaches throughout design stages, regardless of completeness of design model.
4. Provide operative information with minimum user effort.
5. Implement a first principle-based rendering technique that handles high performance building designs well, and produce simulation results within reasonable time constraints.

By meeting these objectives, the new lighting design tool is able to automate much of the previously manual, time-consuming, and disparate efforts in lighting simulation, thus reducing the effort and time-cost. By sharing interoperable information with other tools across the design team, the new lighting design tool is *integrated*. The new tool is also *scalable* in being able to support models of varying completeness throughout all design stages.

[3] See literature review of research on development of performance modeling tools (Chapter 2.1.1), industry surveys on use of simulation tools (Chapter 2.1.2), and assessment of lighting simulation tools (Chapter 2.1.3).

[4] All the reviewed research (Footnote 1, Chapter 2.1.1, Chapter 2.1.2, and 2.1.3) highlight the time-consuming and difficult nature of manually preparing simulation inputs from non-interoperable data, and translating the simulation results to operative information. Research on the development of performance modeling tools (Chapter 2.1.1) and assessment of lighting simulation tools (Chapter 2.1.3) state this effort and time-cost issue explicitly, whereas industry surveys (Chapter 2.1.2) do so implicitly via the respondents' opinions that modeling involves cost and efforts beyond project budgets and have output that are difficult to interpret and apply in design decision making, as well as the respondents' interest in automated code checking technology.

[5] A complete model is defined in this research as one where all attributes, as defined and required for the calculation of performance metrics, and lighting simulation, are explicitly defined.

Keywords: lighting simulation, simulation time-cost, interoperability, automation

Dedication

In loving memory of Russell Huang, 31 July 2009.

Acknowledgements

Special thanks to my committee for their incredible patience, mentoring, and help over the years. Prof. Lam, for going miles beyond teaching and supervising, and the funding that makes this possible. Prof. Akin, for his insights and rigor across an amazing breadth of disciplines. Prof James, for bringing me back to the language of philosophy and science, and keeping me on track by eyeing "the lowest hanging fruit".

I would also like to thank Prof. Ramesh Krishnamurti for his advice, especially on geometric computations.

A preliminary version of the new tool was tested in the NIST ATP research project "Integrated Concurrent Design of High Performance Buildings", conducted jointly by the Center for Building Performance and Diagnostics, Carnegie Mellon University, and United Technologies Research Center.

Contents

Figures

Tables

1. Background and Motivation

Lighting simulation contributes readily to the design of high performance buildings, but contemporary tools are difficult to use, and require prohibitive amounts of time and effort. In addition, existing tools are not able to compute the lighting conditions typical in high performance building designs.

1.1. Introduction

This research is interested in reducing the time and effort to conduct lighting simulation in support of synthesizing high performance building designs. Lighting simulation in this context is more than using a computer program to predict illuminance and luminance of architectural scenes, but entail the entire design context from managing information flows between different domains and tools to defining useful information for design decisions. This research thus falls within the larger research topic of computational design support tools, borrowing on the hypothesis that fast and easy to use tools are beneficial to building design as they help a designer achieve more design iterations within the same design time constraints, given the ability to evaluate design performances much faster than manual processes. The objective of this research is thus to develop such a *fast and easy to use* tool, and the same premise sets the limitation; this research does not purport to investigate the relationship between design quality and design support tools.

Like most architectural endeavors, this research is *integrative* in nature; the new tool developed is an integration of potential technologies and concepts from various domains. While work has been done to extend and modify ideas to suit the architecture lighting design domain, and to achieve the desired reduction in simulation workflow time and effort, this research does not claim authorship to the underlying concepts or even research novelty. Building Information Modeling (Chapter 2.1.3) is a well-established albeit far from ideally implemented concept, and there is has already been much research in Photon Mapping and optimizations (Chapter 4). This research however, posits that a new interoperable design support tool that provides automatic simulation, post-processing, and benchmark calculations, achieved by integrating and adapting such existing technologies, will be fast and easy to use. The focus of this research will thus be on working towards a prototype of this new tool; the validation of the technologies, such as the robustness of building information models, or the specific accuracy of photon mapping techniques, are beyond the scope of this research and will be undertaken in future research. Similarly, while the new tool will be tested against baselines established in this research, extensive user testing is beyond the scope of this research.

The case for reducing time and effort to conduct lighting simulation will be made in this chapter, with observations on pertinent issues. The context of high performance buildings and integrated design processes is also explained. The research objectives for a new integrated scalable lighting simulation tool are then set forth. Chapter 2 (Literature Review) covers existing research and details of fundamental concepts crucial to the development of the new tool, while Chapter 3 (Case Studies of Contemporary Practice) establishes quantitative baselines for time and effort for the research objectives.

Chapter 4 (Implementation of Lighting Simulation Engine) explains the choice of Photon Mapping over RADIANCE as suitable for simulation of high performance building designs, and implementation details. Following principles well established in computer graphics research, the original (Jensen) technique is modified by reducing the degree of approximations and including importance sampling, to maintain first principle-based approaches desired in building lighting simulation.

Chapter 5 (Computational Formulation of Lighting Benchmarks) provides the basis for automatic calculation of typical lighting benchmarks by formulating LEED Credits EQ 8.1 and 8.2 calculation procedures as computational.

Chapter 6 (A Scalable and Integrated Lighting Simulation Tool) describes the software design and implementation details of the new tool, explaining how it achieves the research objectives and overall goal of reducing time and effort in conducting lighting simulation. The new tool is demonstrated in the adaptive-iterative design process, and compared against the time and effort baselines established in Chapter 3.

Chapter 7 (Conclusions) summarizes the work done in this research and highlights important research issues yet to be resolved, most notably simulation engine validation and user testing.

1.2. Background

With studies showing that electric lighting consumes 19% of total global electricity consumption (International Energy Agency, 2006) and lighting conditions affecting occupants' learning capabilities, working performance, absenteeism and user satisfaction (Heshong L. , et al., 2002; Fisk, 2000; Hedge, Sims, & Becker, 1995), lighting design is widely accepted as an important and integral part of the building design process.

Lighting simulation contribute readily to the synthesis of high performance lighting designs. During the design process, the graphical visualization of the predicted lighting environment as afforded by simulation facilitates inductive understanding of the performance of particular designs. This helps the designer achieve better solutions through the typical adaptive-iterative design process. At the same time, metrics of performance can be evaluated efficiently by computational tools, i.e., lighting simulation. Since such metrics are often the operative information required for design decisions, the timely availability of such information speeds up the entire design process.

The current emphasis of a performance-based approach to achieve quality building design entails integrated building design processes whereby multiple disciplines are involved concurrently in the synthesis of building designs. Domain goals are seldom formulated and addressed independently; interim objectives are usually defined following some influence from other domains to achieve some optimal integrated solution. The interim results in turn influence other domain objectives in an iterative manner. It is thus important for lighting simulation tools to adapt effectively to a multi-domain process.

1.2.1. Interoperability

While contemporary lighting simulation tools can (mostly, see 1.2.5 below) accurately predict the performance of lighting designs, it takes a lot of time and effort to conduct lighting simulation. Users typically have to navigate a host of software in preparing and conducting lighting simulation (Figure 1-1). Design information, in the form of CAD drawings, annotations, and documentation in text and spreadsheet formats are incompatible with simulation tools, and have to be manually reconstructed.

This entails a prohibitively high cost in terms of the cost of software, as well as the amount of training required to use them. Duplicate or related information have to be manually defined at each step of the software chain[6], as well as in different domains across the cross-disciplinary design team[7]. Expensive (in terms of both cost and time) error-checking activities, often manual, are necessary to ensure consistency of information between the different software tools.

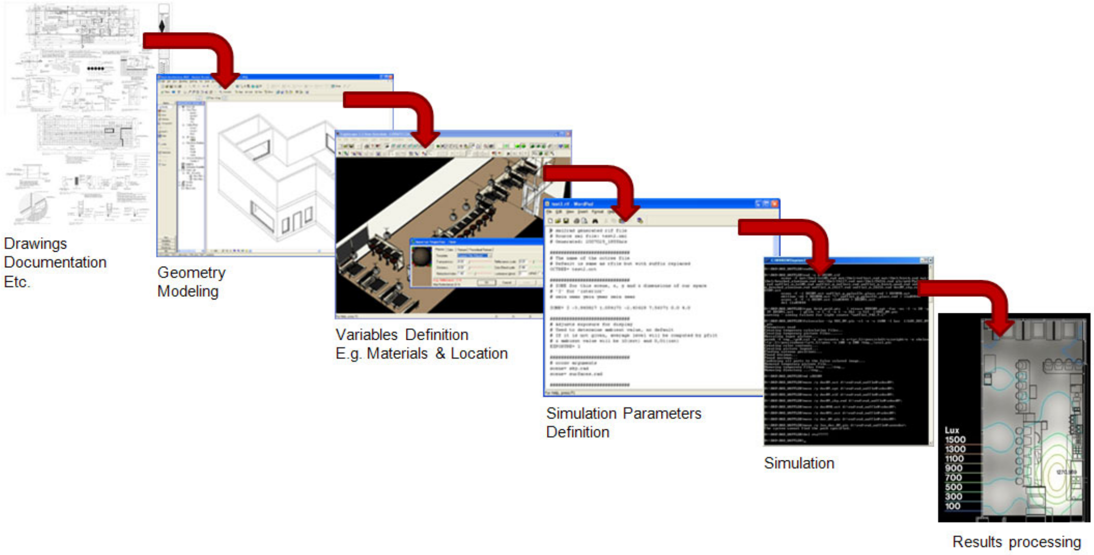


Figure 1-1 Manual definition of duplicate information in lighting simulation preparation

The same problems also exist at the higher level of concurrent processes across different domains; duplicate modeling efforts exist across different domain activities and domain specific assumptions and information definitions often conflict across domains. However, differing semantics in the disparate disciplines, non-interoperable tools and datasets, as well as the difficulty in accessing tacit expert knowledge across domains pose significant challenges to the integration of cross-domain collaborations.

[6] Each software in the typical simulation preparation process deals with specific parts of the overall simulation model, requiring manual checking to ensure consistency. As an example, the method-of-modeling, geometric, construction, and use-schedule information of each architectural element have to be separately and manually redefined using different software (Figure 1-1) in the preparation of a single lighting simulation model.

[7] The taxonomical differences across domains result in much manual translation and error-checking work. As an example, material reflectance properties such as reflectivity described in the lighting domain are indirectly related to roughness, thermal-, solar-, and visible-absorptance descriptions of thermal property in the energy domain.

Currently, the only form of interoperability is limited to syntactical translations between proprietary file formats. CAD tools such as Ecotect (Autodesk, 2009) and Green Building Studio (Autodesk, 2009) are able to export their proprietary models into simulation input files automatically. This is still far from the ideals of seamless Building Information Modeling (BIM); users are still required to manually re-model the entire building design and specify all construction and location parameters in the proprietary tools, as well as spend significant effort on consistency and error-checking. However, the difficulty of learning and managing simulation tools input is almost entirely avoided[8]. While it typically takes months to learn using simulation tools and years to master it, this automatic translation provided by these CAD tools allow architects to treat simulation work as a black-box, with all inputs already taken care of. This limited bridging between CAD and simulation tools proves to be a step in the right direction, and makes simulations much more accessible to the average architect.

Demonstrative of the benefits of interoperability, Ecotect and Green Building Studio are used in the architecture curriculum here in Carnegie Mellon to introduce architecture students (both under-graduate and graduate) to building performance and simulation. In the case of Ecotect, students re-model architectural designs (from other CAD tools) in the software, re-specify and check construction and location properties, and use the automatic export feature to generate RADIANCE (Lawrence Berkeley National Laboratory, 2008) input files for lighting simulation and analysis. The design-orientated user interface, wide range of results visualization tools, such as false-color and model overlay features, prove to be very useful and easy to learn; under-graduate students are able to learn and use the tool within a month, to conduct meaningful lighting analysis and design improvements. The tool also includes automatic building regulation checking (UK Part-L) which might prove useful, but is not tested as the regulations are not intended for the United States.

Green Building Studio, on the other hand, exemplifies another potential of BIM in improving interoperability. The tool imports models from a CAD tool via the gbXML data format (GeoPraxis

[8] In this example, users re-model building designs in Ecotect or Revit (for Green Building Studio use). Assuming correctness in translation into simulation input files, users then do not have to learn how to prepare inputs for the simulation tools, which is typically difficult. However, if there is a need to inspect or modify inputs, then users cannot avoid the relevant parts of the simulation tool. This statement obviously excludes learning the qualitative aspects of simulation tool-use, such as knowledge of when to use, and what tools to use.

Inc., 2003), automatically populates the model with energy-related parameters, and exports energy simulation input files (or conducts the simulation as a web-based service). In a graduate class, students prepare building models in Revit (Autodesk, 2009), a parametric BIM tool. Besides the typical (in architecture practice) geometric and construction modeling tasks, additional thermal zoning information had to be specified. Green Building Studio is then used to automatically populate this building model with all parameters required for energy simulation in EnergyPlus (Lawrence Berkeley National Laboratory, 2008), and generates the simulation input files. Like the above case of making lighting simulation accessible, students are then able to conduct energy simulation without having to spend lots of time learning the difficult and complex simulation tool. While Green Building Studio offers energy rather than lighting simulation, the strategy of automatic parameters population can be applied to the lighting domain.

Data Exchange Formats – gbXML and IFC

While BIM describes the potential for interoperability by capturing all informational aspects of a building throughout its entire lifecycle within a single Building Information Model, the implementation of Building Information Models, and the exchange of such models between tools, requires some form of data format specification. Currently, the Industry Foundation Class (IFC) (International Alliance for Interoperability, 2007), and gbXML (GeoPraxis Inc., 2003), are two prevalent data exchange formats in the building industry. The IFC schema is defined using the EXPRESS data modeling language (IFC2x3) as well as in XML (ifcXML2x3). The gbXML schema is only defined in XML (gbXML 0.37).

There are significant differences between the IFC and gbXML schemas, including comprehensiveness, efficiency, robustness, redundancies, and portability. However, the debate over which schema is better as a data exchange format for Building Information Models is beyond the scope of this research. In terms of comprehensiveness, both formats are not yet able to represent all information across all building performance domains. However, both formats are extensible and can potentially do so (though gbXML was originally developed to capture only information for energy analysis). There are on-going efforts in both cases, in a variety of domains, to extend the schemas in representing more information.

An important and pragmatic consideration in the discussion of data exchange formats is the prevalence of adoption and implementation by all stakeholders in the building industry. Both IFC and gbXML have been implemented by popular architectural CAD vendors, including Autodesk, Bentley and Graphisoft. However, simulation tools, such as EnergyPlus for energy and RADIANCE for lighting, still operate on independent formats. To achieve interoperability between CAD and simulation tools, data in either IFC or gbXML formats in the CAD tools have to be translated into simulation input formats. This can be done either as a feature within CAD tools, or by some middleware[9]. Both strategies are currently used in ongoing efforts[10] to attempt interoperability with energy and lighting simulation tools via both IFC and gbXML; neither format has achieved dominance.

Popular CAD tools such as Revit, Microstation, and ArchiCAD exports building models in both IFC and gbXML formats, which can then be translated into simulation input files by middleware such as Ecotect (lighting and energy simulation input from IFC and gbXML files), Green Building Studio (energy via gbXML only), and the IFCtoIDF tool (energy via IFC only). IES also allows access to both lighting and energy simulation from within the Revit tool, but it uses the gbXML format exclusively for data exchange. A brief discussion of Building Information Models is presented in literature review (2.1.3); this research adopts the gbXML schema to achieve interoperability (6.2.1).

1.2.2. Integrated Building Design Processes and High Performance Buildings

In 2008, buildings (residential and commercial) accounted for 39.7% of U.S. primary energy consumption (U.S. Department of Energy, 2009). With increasing concern over energy use and environment impacts of buildings, as well as the effect of buildings on occupant productivity and

[9] As opposed to software as satisfying some user need (CAD software models building designs, simulation software predicts performance), middleware is only concerned with connecting different software and does not serve any user-need directly (to model a building and analyze its performance). The translation from well-formed IFC or gbXML models into simulation input formats is only syntactical, and does not include any user or computational inputs, data processing, or data manipulation. As such, computer tools that perform only this task are deemed middleware; they only act as intermediaries between software (CAD tool and simulation tool), and does not perform any computation addressing user-needs.

[10] As discussed earlier, both schemas are not yet comprehensive and still being extended. The gaps between model exports and simulation requirements results in less-than-perfect translation.

organizational efficiencies (US Green Building Council, 2002), there is an increasing demand[11] to increase the overall efficiency and quality of buildings; to *maximize operational energy savings; improve comfort, health, and safety of occupants and visitors; and limit detrimental effects on the environment* (City of New York Department of Design and Construction, April, 1999).

High performance buildings refer to buildings with exceptional performance, as compared to typical buildings that only minimally meet budget, time scheduling, functionality requirements, safety regulations and energy codes. High performance buildings achieve exceptional performance in all performance aspects[12], over the entire lifecycle, by optimizing efficiencies across all disciplines (Energy Independence and Security Act of 2007). The emphasis on *performance* in designing high performance buildings is also termed *performance-based* approach.

Integrated building design processes are widely recognized to be necessary in delivering high performance buildings (and corresponding, performance-based approach). In integrated design, multi-disciplinary design teams consisting of all stakeholders[13] are formed, and the entire team participates in all phases of the building delivery process. As compared to traditional design processes comprising discrete and sequential sets of activities by each stakeholder responsible only for their respective well-defined scopes of responsibilities, integrated design processes utilize design charrettes throughout all design stages where the entire multi-disciplinary design team formulate performance objectives, investigate strategies, and formulate solutions together. By drawing upon the needs and expertise of all stakeholders, integrated design processes facilitate holistic performance goals setting, and synergies between disciplines in problem solving.

[11] As evidenced by the prevalent use of high performance, or green, building rating systems, such as LEED and Energy Star, for private and public projects, as well as federal, state and municipal policies establishing high performance building guidelines and regulations.

[12] Besides functional aspects such as building integrity, indoor air quality, lighting quality, spatial quality, energy efficiency, acoustics and building quality (Center for Total Building Performance, 2003), building performance is concerned with the holistic effects of buildings, including but not limited to: economic, occupant-health, safety, welfare, and productivity benefits of high performance buildings (High Performance Building Congressional Caucus Coalition, 2009).

[13] Stakeholders in typical building projects includes building owners, architects, energy consultants, engineers, proposed tenants, state and local government officials, construction contractors, commissioning agent, and Operations and Maintenance staff.

This multi-disciplinary approach includes not just extensive communication and collaboration to ensure complementary building components and systems, but also integrated performance analyses[14] to achieve synergies between components and systems, and consequently improved efficiencies. Whereas independent energy-saving measures can achieve 30% to 50% reductions, integrated designs can raise the savings to 70% (National Science and Technology Council Committee on Technology, 2008). Such inter-disciplinary synergies have been noted to achieve a multitude of benefits across all domains, resulting in more efficient and cost-effective high performance buildings (Prowler, 2008; National Science and Technology Council Committee on Technology, 2008; U.S. Department of Energy, 2001).

While an integrated performance analysis would ideally evaluate the performance a design solution holistically, contemporary performance analysis tools remain domain specific[15]. As such, performance simulation at each step of the adaptive-iterative design process typically consists of multiple domain-specific simulation runs concurrently. Corresponding, there is a need for simulation tools to be interoperable, in terms of sharing design information and simulation results.

1.2.3. Scalability across design stages

As design solutions develop and evolve they possess varying levels of detail (LOD) and ambiguity. Different facets of information are thus available or refined at different stages of the design process. From the point of view of a building information model schema designed to capture all attributes and parameters necessary for lighting simulation, the model may be incomplete in most design stages. Traditionally, this entails the use of different performance metrics at different stages of design. The metrics, requiring only the available information available at each design stage, are then evaluated by corresponding technical approaches.

[14] Integrated performance analyses refer here to a holistic performance evaluation considering all domains. This typically entails integrating performance simulations in all domains, where inter-dependencies are considered, and the analysis based upon all the results from the multiple disciplines.

[15] A comprehensive list of energy-efficiency related simulation tools in multiple domains, including energy, envelope, HVAC, lighting, IAQ, economics, and water, is available at U.S. Department of Energy – Energy Efficiency & Renewable Energy website (http://apps1.eere.energy.gov/buildings/tools_directory/). While some tools offer "whole building simulation" by covering multiple domains (such as TAS and Ecotect), simulation in each domain is conducted sequentially. Integrated performance analyses still entail independent, isolated, simulation in each domain, followed by collating the results for an integrated analysis.

As an example, the Lumen Method[16] and Glazing Factors[17] are typically used in early design stages where much of the building model remains undefined. Following the availability of surface properties and luminaire positions, the Point-by-Point Method[18] and Daylight Factors[19] can be used to assess the design. Towards the final stages of design where accurate evaluations are desired, illuminance distribution and glare conditions are checked by using global illumination simulation. In other words, contemporary simulation tools do not scale with the development of design solutions. Specific metrics, technical approaches and tools are used at different stages of the building design process.

Most lighting simulation tools are thus only appropriate for specific stages of the building design process and used to evaluate the different performance metrics. Without a consistent set of metrics, meaningful comparisons between the performance of design solutions across different stages of design as well as the overall tracking performance is thus problematic. Conceptually, a scalable tool is defined as being able to be used throughout all design stages regardless of LOD (and consequently model completeness), producing a consistent performance metric at all times to allow performance tracking.

1.2.4. Operative Information

As discussed earlier, the identification and use of performance metrics is central to integrated design processes. As opposed to data and information that are not immediately central to the design problem-at-hand, performance metrics directly quantify the quality of the design solution with respect to the defined objectives. Performance metrics facilitate design decisions and action directly, thus the term *operative* information; they enable subsequent operations.

[16] Also known as zonal cavity method, the Lumen Method is used to estimate the number of luminaires (in a uniform layout) required to meet some work-plane illuminance criteria. Luminaire efficacy, illuminance distribution, and surface reflectance effects, are approximated by coefficients.

[17] Glazing Factors estimate the subjective quantity of daylight within a room, using a ratio of exterior to interior daylight illuminance. This method considers the location, size and material characteristics of room fenestrations, and approximates the impact on day lighting via coefficients.

[18] This method calculates of point illuminance by the inverse square law. Only direct illuminations from point light sources are considered.

[19] Similar to Glazing Factors, Daylight Factors describes the ratio of exterior to interior daylight illuminance. Besides the size of room fenestrations, this method also includes coefficients to account for fenestration view angles, fenestration frame thicknesses, and room surface reflectance.

However, lighting simulation tools such as RADIANCE (Lawrence Berkeley National Laboratory, 2008) and Lightscape Visualization System (Autodesk, 1999) only compute the fundamental radiance and irradiance metrics. Design inquires however tend to operate at a higher level of abstraction. There are thus several additional tasks to the use of simulation. First, a well-defined computable problem has to be formulated by considering context and making relevant assumptions. This includes addressing the earlier discussed problem of scalability where design information required for simulation input is not yet available. Second, the problem has to be decomposed into individual tasks each solvable by lighting simulation. Upon completing simulation, the results have to be analyzed and processed into suitable operative information, in this case some desired performance metric, which is useful for design decision making. The execution of this series of tasks requires much expert and tacit knowledge.

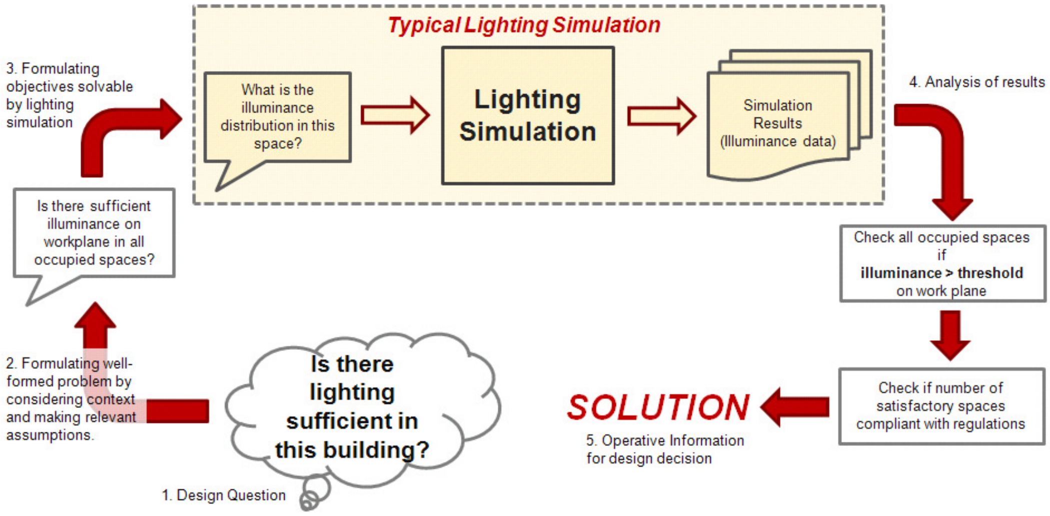

Typical Lighting Simulation
3. Formulating objectives solvable by lighting simulation
What is the illuminance distribution in this space?
Lighting Simulation
Simulation Results (Illuminance data)
4. Analysis of results
Is there sufficient illuminance on workplane in all occupied spaces?
Check all occupied spaces if illuminance > threshold on work plane
2. Formulating well-formed problem by considering context and making relevant assumptions.
Is there lighting sufficient in this building?
SOLUTION
Check if number of satisfactory spaces compliant with regulations
5. Operative Information for design decision
1. Design Question

Figure 1-2 Deriving operative information from simulation

To effectively provide operative information, performance metrics or benchmarks that are most commonly used in design decisions should be identified. The focus of a lighting design support tool would then be to provide such metrics, instead of simply concentrating on lighting simulation.

The United States Green Building Council LEED Rating System (U.S. Green Building Council, 2005) is a popular benchmark in the United States for high performance green buildings and includes two credits for lighting performance: daylight availability and external view availability in building spaces. While the LEED Rating System is a voluntary rating system, the widespread adoption by both governmental and private industry (Landman, 2005) has led to its use as a standard in many building projects. Correspondingly, the consideration of the LEED benchmarks is increasingly a requirement.

These benchmarks are typically calculated only post-design in actual practice due to logistical and resource burdens. Specifically, the procedure for calculating the two lighting benchmarks involve manual processes and data collective that is time-consuming and error-prone. It is noted however, that such benchmarks can potentially serve as performance indicators as the design is being developed; there is much benefit in making the results to these metrics available throughout the design process.

1.2.5. Accuracy and first principle-based rendering techniques

The accuracy of any lighting simulation tool lies in the ability of the tool, via its rendering technique, or algorithm, to solve the global illumination rendering equation. This equation (Kajiya, 1986), first introduced in 1986, describes the complete light transport as energy conserving and is thus physically accurate. The difficulty in solving this equation lies in considering all parts of light transport. Given the recursive nature of diffused irradiance between reflecting surfaces, advanced finite-element algorithms, such as Radiosity, or Monte Carlo methods, such as Monte Carlo (backwards) ray-tracing and photon mapping, have to be used. The different techniques are invariably *estimates* of the rendering equation, each with advantages and limitations in dealing with different aspects of scene geometry conditions and types of light transport; overall accuracy is thus dependent on various factors.

The context of lighting simulation to support high performance building design is used to contextualize accuracy in this research. Buildings are to be used by people; there is no need for accuracy in simulation beyond the human visual threshold of perceivable difference that affects visual task performance. While the issue of visual threshold is complex and dependent upon many factors, Weber's fraction[20] of 0.079 for light intensity serves as a useful metric to the

[20] Weber's law describes human perception to stimuli as logarithmic, and the smallest size of just noticeable difference as a constant proportion of the original stimulus. Weber's fraction of 0.079 for visual stimulus (Teghtsoonian, 1971) means that typical subjects can perceive a lighting intensity variation of 7.9%

degree of accuracy required, where the irradiance and radiance estimates from lighting simulation should be within 8% accuracy. Since an accurate estimate of the rendering equation in the context of buildings is only possible by considering at least light transport mechanisms described by the theory of ray optics (reflection and refraction), rendering techniques that do so are described as *first principle-based.*

In the domain of architectural lighting design, simulation, and research, Monte Carlo (backwards) ray-tracing and Radiosity are the only first principle-based techniques available. Given the small community of lighting simulation and research, there are only two readily-available and prevalent tools: Radiance, from the Building Technologies Program at Lawrence Berkeley National Laboratory (Ward G. J., 1994), and Lightscape Visualization System (Autodesk, 1999), from Lightscape Technologies, Inc. (later acquired and discontinued in 2003 by Autodesk, Inc.). Radiance employs Monte Carlo (backwards) ray-tracing while Lightscape used Radiosity. Other contemporary lighting tools tend to use approximate techniques, such as the split-flux method, that cannot achieve results necessary for high performance lighting design.

The main issue in using Radiosity (as implemented in architecture lighting simulation tools) lies with its limitation in considering only Lambertian (ideal diffused) surfaces. In high performance buildings, common features include daylight re-directors such as light-wells, tubes, and shelves, which are typically glossy. The suitability of Radiosity is thus limited. Similarly, the accuracy of Monte Carlo (backwards) ray-tracing is limited when the light source is a large number of reflections away from the point of interest. As an example, the light arriving on the work surfaces within a high performance building with light re-directors (Figure 4-2) would have gone through multiple reflections (highly-reflected irradiance).

There is thus a need for first principle-based lighting simulation tools for high performance building design, especially those that can effectively evaluate highly-reflected irradiance. While advanced rendering techniques and algorithms have been developed by the computer science community, most of them have been focused on visual presentations, such as for use in movies and games. Accuracy in such cases related more with the visual-cognition issue of photo-realism; they do not readily provide the information relevant to building design.

1.3. Problem Statement

The summary of issues with contemporary lighting simulation tools available for use in building design are:

1. Most of the time in preparing lighting simulations is spent towards the input of existing but non-interoperable information between different tools.
2. Lighting simulation tools do not complement integrated building design processes where the design solution is progressively developed in multiple disciplines concurrently; lighting simulation tools require design information (attributes) that may not yet be defined, and is non-interoperable with other tools.
3. Disparate tools with vastly different technical approaches available for different stages of the building design process do not allow consistent or meaningful performance comparisons between design versions, and similarly makes design performance progress tracking between design versions difficult.
4. Lighting simulation tools provide radiance and irradiance values as simulation results, and much time and manual effort is required to process these results into operative information, information that is directly applicable in making design decisions.
5. Lighting simulation tools employ outdated rendering techniques that are inadequate in evaluating highly-reflected irradiance, a typical feature in high performance building designs.

While there remain other shortcomings in lighting simulation tools as identified by contemporary research, the issues above relate closely to the overall effort and time-cost factors attributed to using simulation tools, which has been consistently identified as obstacles towards using simulation tools. Other issues, notably user-centric issues such as ease-of-use, feedback on accuracy of results, and information processing following simulation results, are not covered by this research.

1.4. Research Objectives

Based on the issues highlighted in the literature review (Chapter 2.1), there seems to be a demand, both academically as well as in the market, for a lighting simulation tool that is interoperable with other tools used within integrated design processes, easy to use, and provides metrics that offer operative information for design decisions. The cost of the tool in terms of training, time and computational requirements must also be affordable.

Considering that contemporary rendering techniques used in building lighting simulations are inadequate in considering typical features found in high performance building designs, a rendering engine following Jensen's Photon Mapping technique (Jensen, 2001) is proposed and implemented.

This research seeks to reduce the effort and time-cost required to conduct lighting simulation by addressing the issues listed above. This goal is structured as the following objectives in a new lighting design support tool:

1. Reduce the time and effort to set up and conduct lighting simulation by using interoperable information from design modeling tools.
2. Complement integrated design processes by supporting design models of varying completeness[21], in a format that is interoperable with tools from other disciplines in the design team. All information, including assumptions, must be consistent across all disciplines.
3. Provide ability to use consistent performance metrics and technical approaches throughout design stages, regardless of completeness of design model.
4. Provide operative information with minimum user effort.
5. Implement a first-principle-based rendering technique that handles high performance building designs well, and produce simulation results within reasonable time constraints.

[21] A complete model is defined in this research as one where all attributes, as defined and required for the evaluation of all solvers in the tool (items 3 and 4), are explicitly defined.

By meeting these objectives, the new lighting design tool is able to automate much of the previously manual, time-consuming, and disparate efforts in lighting simulation, thus reducing the effort and time-cost. To establish a baseline for effort and time-costs case studies of contemporary practice will be conducted later (Chapter 3) and used to measure the new tool.

1.4.1. Interoperability

As discussed earlier, much time and effort in conducting simulation is expended on manually re-modeling and checking non-interoperable data. To meet Objective 1 – Interoperability, the new tool adopts a popular data exchange format and extends it such that it can share Building Information Models with design modeling tools and other domain simulation tools. Assuming a well-formed[22] and complete model from design modeling tools, the new lighting tool would thus have all necessary information to conduct lighting simulation (using the new simulation engine described above); there would be no need for user intervention or manual re-modeling and error-checking. For flexibility and technical comparison, the new lighting tool also supports the RADIANCE simulation engine. Again, all necessary information for simulation is available, albeit organized differently. In this case, an automatic translator is implemented to organize the information into the RADIANCE syntax.

Satisfaction of this objective is measured by the ability of the new lighting tool to achieve effective building information model interoperability with design modeling tools, and automatically preparing lighting simulation inputs. Assuming well-formed and complete models from design modeling tools, the new tool would effectively eliminate all the manual time and effort currently associated with setting up and conducting lighting simulation. Minimally, the interoperable information would include geometry, construction, material, lighting equipment, location, and project information. This research adopts the gbXML schema to achieve interoperability (6.2.1 below).

The current gbXML schema (version 0.37) is extended to represent all information necessary for simulation, and an automatic parser is implemented in the new tool to achieve information interoperability with design modeling tools using this extended schema. The new tool also

[22] A well-formed model is defined in this research as one that is semantically and syntactically correct; it is independent of completeness.

exports data in the same extended schema to further enhance interoperability with other tools. Note that the development of other tools is beyond the scope of this research. Since most prevalent design CAD tools already support data exchange via gbXML, it is assumed that implementing the extended schema and obtaining well-formed models is feasible.

Since the new tool also supports incomplete models (see 1.4.2 below), and the current gbXML is defined as a subset of the extended version, the new tool is also interoperable with design modeling tools supporting the current gbXML schema. In this research, the popular parametric BIM tool Revit (Autodesk, 2009) is used to generate building models and demonstrate interoperability. Like the choice of gbXML, the use of this tool is in the context of demonstrating time and effort savings via interoperability, and not a discussion on the merits of particular commercial products.

1.4.2. Integrated Design Processes

To meet Objective 2 – Complement Integrated Design Processes, the new tool must be able to support design models of varying levels completeness, and share all information across different disciplines and domains in the project team. A complete model is defined in this research as having all the necessary information for lighting simulation, as exemplified by one where all data element-types defined as required-types within the extended gbXML schema are instantiated. Nominally, this describes a model with geometry, construction, material, location, and project information[23]. As an example, early schematic designs might not include construction and material details beyond generic descriptions, and is thus *incomplete*. Even when such information becomes progressively available as design progresses, material reflectance properties necessary for lighting simulation are not available; assumptions have to be made within the lighting domain, independent of other domain considerations. Likewise, location information such as sky descriptions and project information such as camera-views for analysis tend to be lighting-domain specific assumptions.

Satisfaction of this objective is measured by the ability of the new lighting tool to still automatically prepare lighting simulation inputs (Objective 3 above) for models of varying levels

[23] Lighting equipment is not defined as a required type since, while not common, it is theoretically possible that a building design does not include any electric lighting. The more common occurrence of this situation is when all electric lighting is omitted to consider the day-lighting effects in isolation.

of completeness, and eliminate the need to manually or explicitly check for consistency in any assumptions made. The need to share interoperable Building Information Models with tools across the design team is implicitly met upon satisfaction of Objective 1 above via data export in the extended gbXML data exchange format.

The new tool implements two separate strategies to meet the two-fold objective. By observing general building typologies and associated construction and material use within United States, heuristic rule-sets are used to populate missing information in the models to achieve completeness, and global (project-wide) data-sets are used to ensure consistent assumptions across different disciplines and domains in the project team. Following the same considerations as above for interoperable Building Information Models, both rule-sets and data-sets are implemented in XML formats.

1.4.3. Scalability- Consistency across Design Stages

To meet Objective 3 – Consistent Metrics and Approaches, the new tool is designed to use the automatic benchmark calculators and rendering technique (described later) throughout all design stages. Satisfaction of this objective is measured by the ability of the new lighting tool to perform consistently regardless of design model completeness. The same benchmark calculation and rendering techniques should be used, and the same operative information provided, under all conditions.

Since the tool maintains a complete model at all times as a strategy to deal with design models of varying completeness (Objective 2 above), it is now possible to consistently use the new rendering engine throughout all design stages. As implied in later discussion on the new rendering technique and scalability (1.4.5), it is reasonable to expect varying time-constraint, results-accuracy, and even visualization-bias, at different design stages. While these issues are beyond the scope of this research, they are invariably part of the scalability consideration. As discussed later, the new rendering technique accommodates progressive calculation, and offers a fast-visualization mode. With regards to the overall goal of reducing effort and time-cost required to conduct lighting simulation, the inclusion of an internal simulation engine further avoids the earlier discussed problem of disparate tasks and tools associated with simulation work.

Similarly, the maintenance of a complete model at all times allows the LEED benchmarks to be calculated throughout all design stages. The implemented algorithms compute the benchmarks within seconds, though possible optimizations and opportunities to implement progressive result refinement are identified.

In keeping with the overall goal of reducing the effort and time-cost required to conduct lighting simulation, the GUI of the new tool is designed to require minimal user intervention, while all parameters can still be inspected and edited if desired. In typical use, there is no need for *any* user intervention besides selecting an input design model; the new tool automatically inspects the input design model, populates all necessary and missing parameters, calculates the two LEED performance benchmarks, and runs a lighting simulation using the new rendering technique to obtain point-luminance values. The new tool is thus *scalable* in being applicable, and uses consistent approaches, throughout all design stages.

1.4.4. Automated Calculation of Performance Benchmarks

To meet Objective 4 – Provide Operative Information, two commonly used lighting performance benchmarks from the LEED Rating System are formulated as computable. They are implemented as automatic calculators so that no user intervention is required. Satisfaction of this objective is measured by the ability of the new lighting tool to automatically calculate the benchmarks regardless of the stage of design, and the benchmarks updated dynamically whenever there are changes made to the design model.

To be able to calculate the two LEED benchmarks, they are first formulated as computable, with all necessary parameters identified and included within the BIM to be used by the new tool (1.4.2 above). Algorithms are then designed and implemented to automatically calculate the two benchmarks. By using a finite-element approach and considering the degree of accuracy required in building design, the benchmarks are computed within seconds, and can be updated dynamically during adaptive-iterative design activities.

Besides achieving automated calculations for LEED, the tool also reduces documentation time by providing the relevant data in LEED submittal formats. This further contributes to the reduction in project time and effort.

1.4.5. First Principle-based Simulation Engine

To meet Objective 5 – First Principle-based Rendering Engine, a new rendering engine following Jensen's Photon Mapping technique is implemented. Satisfaction of this objective is measured by the ability of the new rendering engine to handle lighting conditions typical in high performance design features.

The Photon Mapping technique is chosen for its ability to be able to evaluate highly reflected irradiance typical in high performance design features, as opposed to the Monte Carlo (backwards) ray tracing, which requires significantly more time and resources to do so. Considering the building lighting analysis-use context and principles well established in computer graphics research, modifications are made to the classic (Jensen) Photon Mapping technique. The modifications include reducing the degree of approximation in irradiance estimates, using constant sampling areas and area corrections when using Photon Maps, and importance sampling via a power-prioritized sub-sampling technique to build Photon Maps efficiently.

The implementation of the new technique in the new lighting tool also considers the issues of varying time constraints, especially in the context of scalability. Since the new simulation engine is used by the new tool for all design stages (1.4.3 above), it will be subjected to varying time constraints. While the survey, quantification, and generalization, of time constraints at various stages of design is beyond the scope of this research, the new simulation engine is progressive; it can generate base results, and then improve the quality of the results incrementally with more time. This approach facilitates future research and optimization with respect to time-constraints and level of accuracy required.

A common problem with the Photon Mapping technique is the amount of time it takes to generate the final image; empirical tests in this research shows that it takes upwards of 16 hours to generate a high quality solution when the lighting distribution has extremely high variability and the steradians subtended by the radiance peaks as seen by most of the points in the scene are small. By considering the quantitative and qualitative qualities of the solution as two separate issues, and taking into consideration the desired level of accuracy of each in the context of building design, the feature to directly visualize the Photon Map is implemented so that the simulation engine can yield the required solutions within a significantly lower amount of time. In the same empirical test mentioned, the time was reduced to a mere 6 minutes. While the solution is no longer photo-realistic, it is still sufficiently accurate in terms of lighting distribution, physical radiance and irradiance values, as well as cognitively representative of the design conditions.

2. Literature Review

Research shows that the computational performance modeling tools are difficult and time-consuming to use, and have not improved much over the last two decades. Radiance, which uses Monte Carlo (backwards) ray tracing, is commonly accepted as the most accurate lighting tool available. The fundamentals of lighting physics, illumination models, and the rendering equation, are also reviewed.

2.1. Performance Modeling Tools

Much research and use of performance modeling tools within the architecture design process is focused on the domain of energy use. While this domain is different from that of lighting, the generalization of using computational support for a performance-based approach to architecture design allows the insight to be relevant to all performance modeling tools in general. Furthermore, there is also a close correspondence between the nature of energy and lighting modeling in terms of the information requirements, modeling parameters, algorithms, and resources required.

2.1.1. Development of performance modeling tools

In reviewing the development and use of simulation tools in North America and Europe (Augenbroe & Winkelmann, 1991), Augenbroe pointed out that only 10% of the building industry used performance modeling tools. The reasons for such low levels of use included:

1. The tools were difficult to learn, frustrating to learn many tools
2. Inputs, particularly geometric information, were difficult and time consuming to prepare
3. Outputs difficult to interpret, requires expert knowledge to translate to design information
4. Difficult to ascertain level of accuracy

18 years later, many of the issues remained unanswered. While there is now a myriad of modeling tools, they remain domain specific, post-design, evaluation tools used by specialists.

Even the same author notes after a decade (Augenbroe, 2001) that most of the previously listed complaints remain. Modeling tools were still difficult to learn, not user-friendly, had limited results presentation, and could not validate assumptions. The same paper highlights additional features expected of contemporary performance modeling tools including functionalities to support rapid parametric studies, and feedback on the accuracy of results.

The use of modeling tools still requires large amounts of effort and time for manual data input, a tedious and error prone task that consumes almost half the total time required for conducting simulations (Vladimir, 2001), with up to 80% of this time spent on defining the building geometry. Only a very small amount of time is actually required for computation; the remaining half of the total effort is spent on analyzing the results. This confirms that acquisition geometry (and other common) data and the migration of physical knowledge from modeling tools into meaningful design information remains two key obstacles to the ease of using modeling tools.

2.1.2. Industry Surveys

An industry survey conducted (Wong, Lam, & Feriadi, 1999) revealed that only 11% of the architecture firms surveyed used lighting software, and predominantly for enhancing the visual impression of the design rather than quantitative assessments. The survey respondents reflected an emphasis on first costs by the clients; they were of the opinion that modeling involves extra cost and effort beyond project budgets. A majority of the respondents felt that modeling tools were expensive to maintain and upgrade, requires large amounts of data input that may not be available at the time of design, were largely platform dependent, and that modeling output could be difficult to interpret and apply in design decision making. Half of the respondents also felt that modeling tools are not user-friendly and difficult to learn.

A survey of architects conducted as part of a research project assessing energy modeling tools (Lam, Huang, & Zhai, 2004) revealed similar results with the previous studies and survey. From the feedback, modeling tools were felt to be lacking in terms of data interoperability with other prevalent industry software, parametric analysis features, feedback on the accuracy of results, and results post-processing into design-decision relevant information. However, the recent implementation of the LEED building rating system has generated some interest in using performance modeling tools.

A recent survey by (McGraw Hill, 2007) shows that 28% of building firms in the United States use Building Information Models (BIMs), with the top 5 reasons for doing so as:

1. Less time re-entering data manually
2. Requirement from clients
3. Improved communication among stakeholders
4. Ease of parametric modification to designs
5. Opportunity to reduce costs

Based on current trends, the survey estimates 38% of firms to use BIMs by 2008 and 49% in 2009. While the adopting of BIMs is clearly cost driven as it has demonstrated reduction of repetitive tasks, the respondents have also expressed interest (85% of architects and 42% of engineers) in automated code checking technology. Currently only 13% of build teams have tried such technology. On average, architects spend 49 hours per project on code checking, engineers 52 hours.

2.1.3. Building Information Models and Interoperability

Building Information Models, as the name implies, are digital representations of a building containing information required to facilitate activities in the building delivery process, from design (program and intent specifications, design-schemes evaluations and visualizations, design specifications), construction (construction contracts, procurement, and schedules, clash detection, building commissioning, as-built documentation), to management (facilities management, post-occupancy commissioning, systems controls). Given the myriad activities, it is obvious that there is:

1) a corresponding multitude of software involved in the building delivery process

2) a large set of information required.

While building delivery, especially integrated design processes (1.2.2), would ideally utilize a single platform, software, and information model for all activities, the (contemporary) development and implementation costs for such platforms and software are obviously prohibitive. This emphasizes the importance and benefits of BIMs, as observed through the industry survey presented in the preceding section. The objectives of BIMs, as pertinent to different stages of building design (Eastman, Teicholz, Sacks, & Liston, 2008), are presented below.

Pre-construction

1. Capturing building owner intents and requirements
2. Evaluating and tracking performance of proposed schemes from schematic stage

Design

3. Earlier and more accurate visualizations of designs
4. Automatic low-level corrections when changes are made to design
5. Generate accurate and consistent drawings for different disciplines at any stage of design
6. Earlier collaboration of multiple design disciplines
7. Earlier check with design intent
8. Exact cost estimates
9. Accurate energy and sustainability evaluations

Construction and Fabrication

10. Synchronize design, procurement, site management, and construction activities
11. Clash detection before construction
12. Evaluation and documentation of design and construction changes
13. Generation of fabrication models

Post-Construction

14. Commissioning specifications and documentation
15. Facilities management and operations models

Table 2-1 Benefits of Building Information Models (Eastman, Teicholz, Sacks, & Liston, 2008)

To achieve the above objectives, BIMs are more than a combination of traditional (digital) documents such as 2D drawings, 3D models, or spreadsheets. BIMs describe buildings parametrically, with non-redundant specification of objects, attributes, and relationships such that the model can be consistently viewed at different levels of detail and from different viewpoints (for different disciplines). This facilitates the collaborative and integrated design intent by allowing the information within the model to be consistently and automatically transformed into any format as required by the diverse disciplines and activities.

To achieve accessibility and durability in the context of multiple tasks and software involved in the building delivery process, BIMs should be interoperable between all the involved software, and there is a need to manage the access, changes, and updates to the BIMs by various activities. Currently, there are various on-going efforts and developments in data formats (to enable interoperability) and data repositories (to facilitate change management); there is yet to be predominant or standardized use of either within the building industry.

While there is a wide range of data formats to transfer information between software, notable formats that are public (open source), and support Building Information modeling (non-redundant parametric object-based models), are:

1. Industry Foundation Classes (IFC2x3) (International Alliance for Interoperability, 2007)
2. ifcXML2x3 (International Alliance for Interoperability, 2008)
3. aecXML (FIATECH, 2007)
4. gbXML (GeoPraxis Inc., 2003)

ifxXML2x3 is simply the IFC schema implemented in XML (International Alliance for Interoperability, 2008a), while the aecXML effort has been subsumed under the IFC effort since May 2003 (version IFC2x2). As noted earlier (0), the most prevalent data formats, as adopted by both design modeling and (lighting and energy) simulation tools, are IFC and gbXML. IFC data models are expressed using the ISO-STEP EXPRESS language while gbXML uses XML. Although ifcXML also uses XML, it is not widely supported by design modeling tools (Design Computing, Georgia Institute of Technology, 2009). Of the three prevalent building modeling tools (Revit, Microstation, and ArchiCAD), only ArchiCAD supports ifcXML.

The IFC schema (Figure 2-1) is an extensible, object-based, framework of hierarchical entities designed to represent all types of building information throughout the entire building lifecycle. Basic entity objects (Figure 2-1 Resource Layer) are used to define fundamental elements and properties, which are then inherited by incrementally higher level entities (Figure 2-1 Core, Interoperable, then Domain Layers) to describe complex concepts, types, objects, or features. With each level of inheritance, new attributes and properties are added to cumulatively form the overall object description; the attributes and properties are also incrementally defined in the same manner. The comprehensiveness of the IFC schema, and constructible ontological sets, is thus limited only by the set of base entities, types, and enumeration. Given some 342 basic entities within 26 categories in the fundamental Resource Layer, the IFC schema is potentially able to represent a wide range of information. The IFC schema contains general purpose

IfcPropertyDefinition, IfcPropertySet, and *IfcRelationship* entities to allow definition of objects that are not currently available, thus allowing extensibility without modifications to the schema.

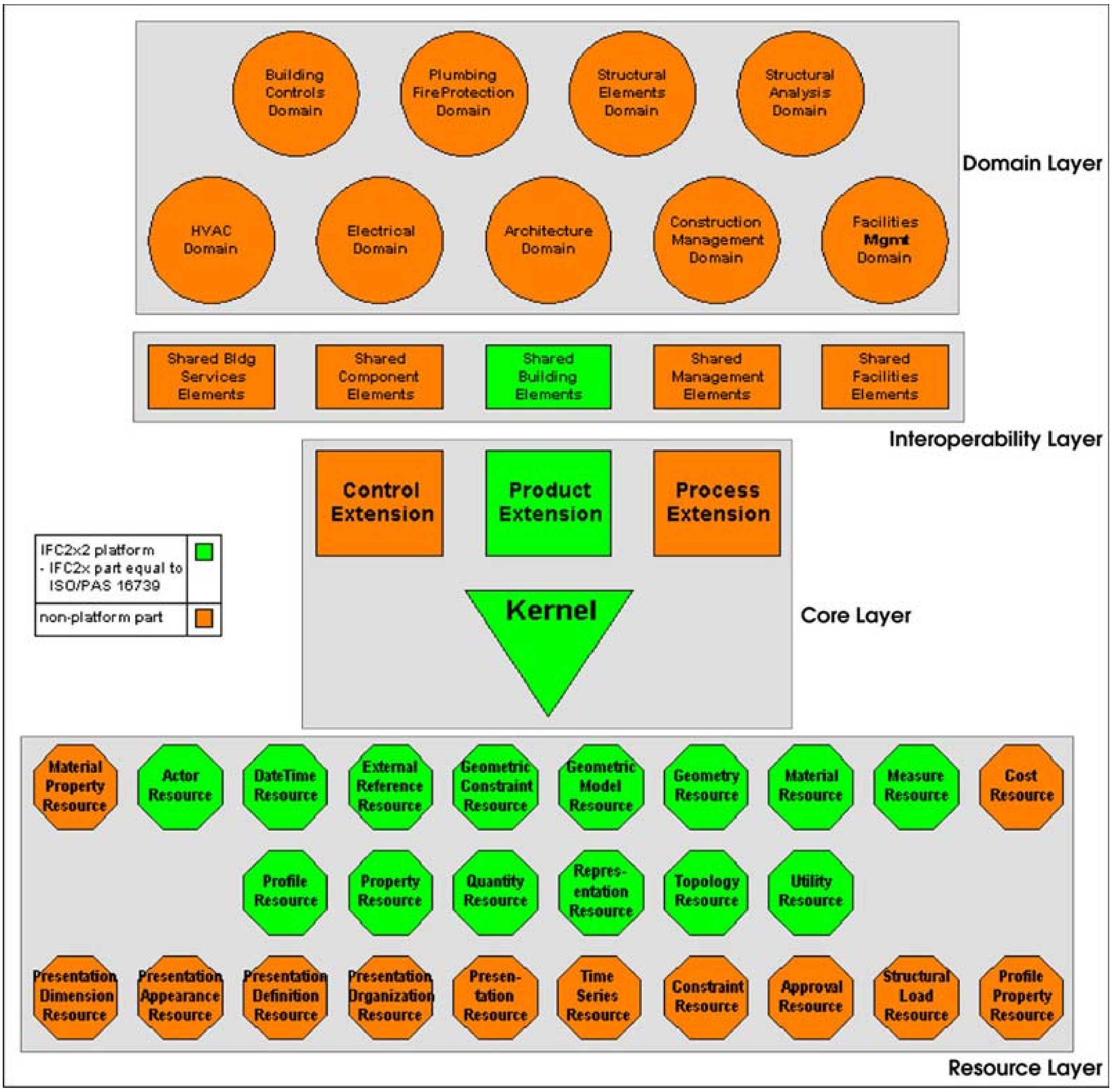


Figure 2-1 IFC Schema (International Alliance for Interoperability, 2007)

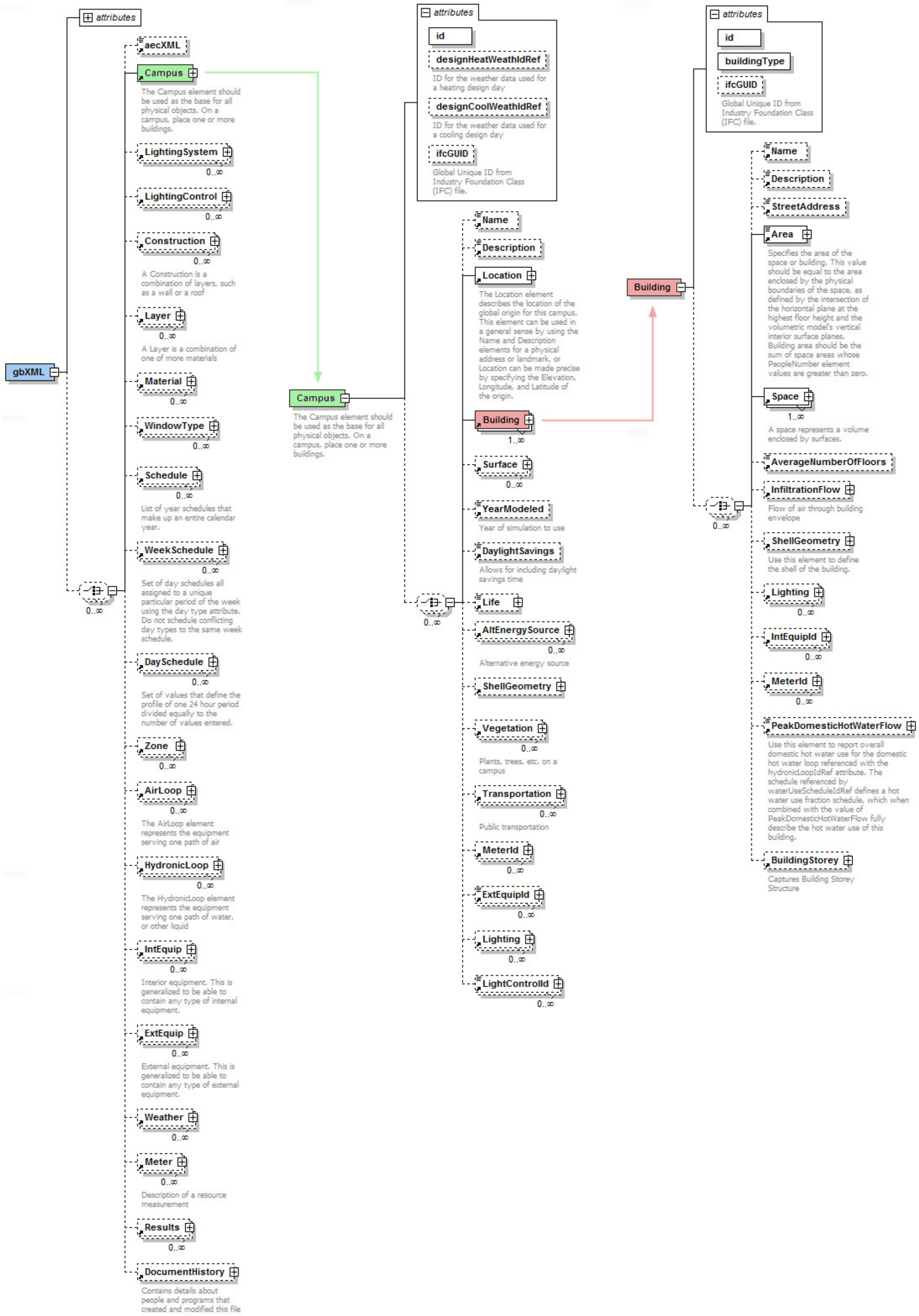

attributes
gbXML
0..∞
aecXML
Campus
The Campus element should be used as the base for all physical objects. On a campus, place one or more buildings.
LightingSystem
0..∞
LightingControl
0..∞
Construction
0..∞
A Construction is a combination of layers, such as a wall or a roof
Layer
0..∞
A Layer is a combination of one of more materials
Material
0..∞
WindowType
0..∞
Schedule
0..∞
List of year schedules that make up an entire calendar year.
WeekSchedule
0..∞
Set of day schedules all assigned to a unique particular period of the week using the day type attribute. Do not schedule conflicting day types to the same week schedule.
DaySchedule
0..∞
Set of values that define the profile of one 24 hour period divided equally to the number of values entered.
Zone
0..∞
AirLoop
0..∞
The AirLoop element represents the equipment serving one path of air
HydronicLoop
0..∞
The HydronicLoop element represents the equipment serving one path of water, or other liquid
IntEquip
0..∞
Interior equipment. This is generalized to be able to contain any type of internal equipment.
ExtEquip
0..∞
External equipment. This is generalized to be able to contain any type of external equipment.
Weather
0..∞
Meter
0..∞
Description of a resource measurement
Results
0..∞
DocumentHistory
Contains details about people and programs that created and modified this file
Campus
The Campus element should be used as the base for all physical objects. On a campus, place one or more buildings.
attributes
id
designHeatWeathIdRef
ID for the weather data used for a heating design day
designCoolWeathIdRef
ID for the weather data used for a cooling design day
ifcGUID
Global Unique ID from Industry Foundation Class (IFC) file.
0..∞
Name
Description
Location
The Location element describes the location of the global origin for this campus. This element can be used in a general sense by using the Name and Description elements for a physical address or landmark, or Location can be made precise by specifying the Elevation, Longitude, and Latitude of the origin.
Building
1..∞
Surface
0..∞
YearModeled
Year of simulation to use
DaylightSavings
Allows for including daylight savings time
Life
AltEnergySource
0..∞
Alternative energy source
ShellGeometry
Vegetation
0..∞
Plants, trees, etc. on a campus
Transportation
0..∞
Public transportation
MeterId
0..∞
ExtEquipId
0..∞
Lighting
0..∞
LightControlId
0..∞
Building
attributes
id
buildingType
ifcGUID
Global Unique ID from Industry Foundation Class (IFC) file.
0..∞
Name
Description
StreetAddress
Area
Specifies the area of the space or building. This value should be equal to the area enclosed by the physical boundaries of the space, as defined by the intersection of the horizontal plane at the highest floor height and the volumetric model's vertical interior surface planes. Building area should be the sum of space areas whose PeopleNumber element values are greater than zero.
Space
1..∞
A space represents a volume enclosed by surfaces.
AverageNumberOfFloors
InfiltrationFlow
Flow of air through building envelope
ShellGeometry
Use this element to define the shell of the building.
Lighting
0..∞
IntEquipId
0..∞
MeterId
0..∞
PeakDomesticHotWaterFlow
Use this element to report overall domestic hot water use for the domestic hot water loop referenced with the hydronicLoopIdRef attribute. The schedule referenced by waterUseScheduleIdRef defines a hot water use fraction schedule, which when combined with the value of PeakDomesticHotWaterFlow fully describe the hot water use of this building.
BuildingStorey
Captures Building Storey Structure

Figure 2-2 gbXML Schema (GeoPraxis Inc., 2003) with ***Campus*** and ***Building*** Elements expanded

In contrast to the generality of the IFC schema, the gbXML (version 0.36) schema (Figure 2-2) uses 285 object types and 99 enumerations to define a static ontological description of building information, focusing on the informational needs of energy and engineering analyses. Like IFC, gbXML is hierarchical, and uses basic entities to form higher level objects, though there is no structured inheritance behavior. By using XML, additional attributes or properties can be introduced by simply inserting element tags in accordance to the XML specification (W3C, 2000).

In considering the informational needs for lighting simulation, both IFC and gbXML schemas currently do not have explicitly defined entities that can be used directly to describe necessary objects, such as sky luminance distribution models, or material Bi-directional reflectance distribution functions (BRDFs). However, both schemas are potentially extensible to do so. A comparison of the two schemas is presented below.

	IFC	gbXML
Modeling Language	EXPRESS	XML
Public (Open Source)	Yes	Yes
Parametric objects	Yes	Yes
Comprehensive (for lighting)[1]	No	No
Location Information	Yes	Yes
Sky Model	No	No
Complex Geometry[2]	Yes	Yes
Material (BRDF for lighting)	No	No
Lighting equipment	Yes	Yes
Lighting distribution	Yes	No
Lighting Simulation Parameters	No	No
Lighting Simulation Results	No	Yes
Extensible[3]	Yes	Yes

[1] Refers to schema having (the following) objects to capture information for lighting simulation

[2] Refers to general polygon surfaces (that might be non-orthogonal)

[3] Refers to extensibility of schema to form new objects, capturing missing information

Table 2-2 Comparison of IFC and gbXML in capturing information for lighting simulation

2.1.4. Lighting Simulation Tools

According to research (Ubbelohde, 1998; Roy, 2000; Bryan & Mohammed Autif, 2002; Estes, Schreppler, & Newsom, 2004), contemporary lighting simulation tools were assessed and found to have the following shortcomings:

- Difficult to use
- Hard to learn, frustrating to learn many tools

- Geometric input tedious and error prone
- Required inputs difficult to obtain
- Modeling limitations
- No feedback on accuracy
- Output difficult to interpret
- Does not support integrated, concurrent design processes
- Difficult to transfer data between domains
- Does not validate assumptions
- Difficult to conduct parametric analysis
- Difficult to transfer findings between domains

2.1.5. RADIANCE as State-of-the-art

In the same research cited above, only RADIANCE (Lawrence Berkeley National Laboratory, 2008), Lightscape (Autodesk, 1999) and Inspirer from Integra Inc. (not distributed in United States) were considered as physically-accurate.

All the tools except Lightscape require geometry modeling from scratch, though the former actually also requires models that are semantically different from typical architectural models. This essentially means that all tools require re-modeling when given a design model. "Simplistic" tools such as Lumen Micro from Lighting Technologies Inc. often achieve user-friendliness at the expense of accuracy, often to the extent of being unsuitable for use in architecture design. At the other extreme, the highly accurate RADIANCE tool was difficult to use, requiring much training and time to use well.

Through empirical and theoretical tests (Kopylov, Khodulev, & Volevich, 1998; Ng, Lam, & Nagakura, 2001; Ruppertsberg & Bloj, 2006), RADIANCE is also deemed the most physically accurate lighting simulation tool available. While this research is not the first to try to introduce Photon Mapping as a technique to overcome the limitations of contemporary building lighting simulation tools, previous research (Schregle, 2005) has only implemented classic (Jensen) Photon Mapping as-is (4.2 below); this research modifies the Photon Mapping technique so that it is more accurate and suitable for building lighting analysis.

2.2. Lighting Physics

While the nature of light is still not completely understood, and corresponding there exists many models of light, ray optics (or geometric optics) is commonly deemed sufficient to model the visual phenomenon of light in the context of architecture. Ray optics assumes that light travels as independent rays in straight lines through optical mediums, and interacts with such mediums according to a set of geometric rules. This model describes most of our daily visual experiences, such as reflections and refractions. While the model assumes light has infinite speed, and achieves steady-state instantaneously. This is not a problem for lighting simulation since the latter is conducted for time-instant scenes, and requires such steady-state assumption for ease of computation. While ray optics does not consider phenomena such as interference, diffraction, polarization, dispersion, or gravitational influences, it is still sufficient for describing architectural scenes, and already difficult to implement in simulation engines.

2.2.1. Radiometry and Photometry

Radiometry units are often used to quantify light. The basic unit is radiant power, also called radiant flux. *Radiant flux,* **Φ**, is the time rate flow of radiant energy expressed in Watt (W) (*Joule/second*).

$$\Phi = \frac{dQ}{dt} \quad \text{[W]}$$

Eq. 2-1

where *Q* is the *Radiant energy* and can be computed by integrating the spectral energy over all possible wavelengths. For the purposes of lighting simulation for architecture lighting, it is not necessary to go beyond the definition of radiant flux since the flux outputs of lamps are usually available (though in photometric units – luminous flux [lumens]). In lighting simulation, radiant flux is usually represented in three separate components(Φ_r, Φ_g, Φ_b), corresponding to the three primaries (trichromatic color vision) red, green and blue within the visible spectrum. Each component integrates the spectral energy over the wavelengths picked up by the corresponding color receptors (cone cells).

The difference between radiometry and photometry is that the latter includes standard human visual responses. While radiant flux includes energy over all wavelengths, luminous flux includes only energy over visible wavelengths. Since lighting simulation is concerned with physical accuracy, radiometric units are used. However, since lighting simulation inputs (radiant flux of luminous sources, trichromatic material properties) are described in terms of their visible properties, radiometric and photometric units are essentially interchangeable in this research.

The *Radiant Intensity,* ***I***, is the radiant flux per differential solid angle, $d\vec{\omega}$:

$$I(\vec{w}) = \frac{d\Phi}{d\vec{\omega}} \quad [\mathrm{W \cdot sr^{-1}}]$$

Eq. 2-2

The *radiant flux area density* is the radiant power per unit surface area. This is usually separated into flux leaving a surface and flux incident on a surface. The solid angle is a three-dimensional measure of angles and analogous to the two-dimensional radian. It describes angular size and is measured in steradians, a SI derived unit. By definition, 1 steradian is the solid angle subtended at the center of a sphere of radius *r* by a portion of the surface of the sphere having an area r^2. This means that steradians are dimensionless ($m^2 \cdot m^{-2}$).

Irradiance, ***E***, is the incident radiant power arriving at a surface location, *x* (differential flux per differential area). The photometric equivalent is *illuminance*.

$$E(x) = \frac{d\Phi}{dA} \quad [\mathrm{W \cdot m^{-2}}]$$

Eq. 2-3

Illuminance is commonly used in building design and regulations to evaluate if there is sufficient light incident upon work surfaces for safe and comfortable performance of visual tasks. Design benchmarks typically recommend ranges of illuminance values, as well as distribution, according to building types and use (The American Society of Heating, Refrigerating and Air-Conditioning Engineers, 1999; Illuminating Engineering Society of North America, 2000; Commission Internationale de l'Eclairage, 2001). Calculating illuminance, or irradiance, is thus one of the objectives of lighting simulation.

Radiant Exitance, ***M***, or *Radiosity,* ***B***, is the exitant radiant power leaving a surface location, *x*:

$$M(x) = B(x) = \frac{d\Phi}{dA} \quad [\text{W}\cdot\text{m}^{-2}]$$

Eq. 2-4

Radiance, ***L***, is the radiant flux per unit solid angle per unit projected area (Figure 2-3). It is indicative of the brightness of objects. The photometric equivalent is *Luminance* (Table 2-3).

$$L(x, \vec{\omega}) = \frac{d^2\Phi}{dA^{\perp} d\vec{\omega}} = \frac{d^2\Phi}{\cos\theta dA d\vec{\omega}} \quad [\text{W}\cdot\text{m}^{-2}\cdot\text{sr}^{-1}]$$

Eq. 2-5

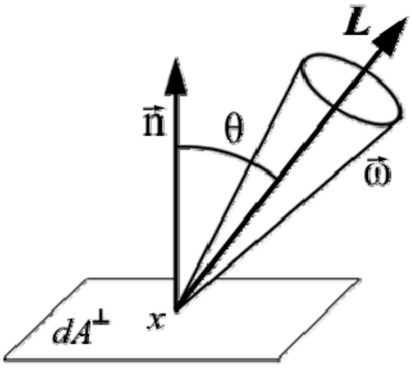


Figure 2-3. Radiance $L = (x, \vec{\omega})$, flux per unit solid angle $\vec{\omega}$, per unit projected area $dA^{\perp}$

Like illuminance, luminance is a common metric used in building design. Since luminance describes the objective brightness of objects, and human vision perceives only limited brightness ranges instantaneously and can be damaged by excessive brightness, luminance values and ranges are used (Commission Internationale de l'Eclairage, 1995) to evaluate if the visual experience is comfortable and conducive to the visual task. Calculating luminance, or radiance, is then the second objective of lighting simulation.

	Radiometry		Photometry	
Source Power	*Radiant flux* $\boldsymbol{\Phi}$	watts	*Luminous Flux* $\boldsymbol{\Phi_v}$	lumens
Light power in given direction	*Radiant Intensity* $\boldsymbol{I}$	watts/sr	*Luminous Intensity* $\boldsymbol{I_v}$	candela (lumens/sr)
Incident light power on surface	*Irradiance* $\boldsymbol{E}$	watts/m^2	*Illuminance* $\boldsymbol{E_v}$	lux (lumen/m^2)
	Radiant Exitance $\boldsymbol{M}$ *Radiosity* $\boldsymbol{B}$	watts/m^2		
Light power per solid angle per projected area	*Radiance* $\boldsymbol{L}$	watts/m^2·sr	*Luminance* $\boldsymbol{L_v}$	cd/ m^2

Table 2-3. Comparison of radiometry and photometry units

To convert between radiometric and photometric units, two types of efficacies have to be considered. The first is luminous source efficacy (lm/W), which translates the power (Watts) of luminous sources into luminous flux (Lumens). The second is the human vision visual response, which accounts for varying sensitivities of human vision to different wavelengths (Figure 2-4)

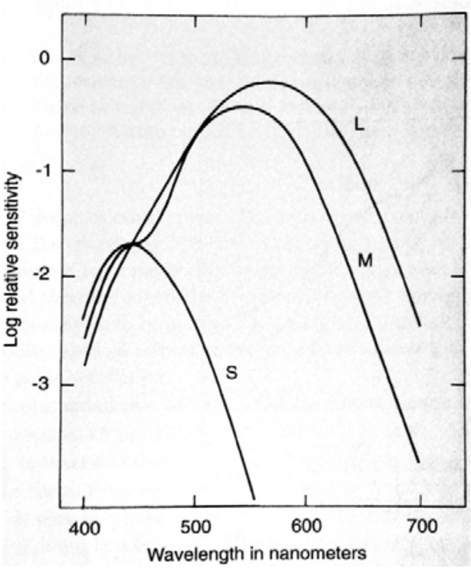


Figure 2-4 Relative spectral sensitivity of long (L), middle (M), and short (S) human cone photoreceptors (Illuminating Engineering Society of North America, 2000)

Fortunately, as mentioned above, the luminous flux (photometric unit) outputs of lamps are typically specified by manufacturers. This avoids the need for simulation tools to determine appropriate luminous source efficacies. Since both the input (luminous flux) and desired results (illuminance and luminance) of lighting simulation are in photometric units, there is no need to worry about the correct conversion factor, as long as a consistent factor is used. Incidentally, standard (photopic) human vision (Commission Internationale de l'Eclairage, 1931) (Figure 2-5) is most sensitive at 555nm, and this corresponds to a peak of 683 lumens per watt from a standard illuminant. Correspondingly, the luminance is defined as a derived SI unit of luminous intensity, measured in candela, which is defined as *"the luminous intensity, in a given direction, of a source that emits monochromatic radiation of frequency 540 x 10^{12} hertz and that has a radiant intensity in that direction of 1/683 watt per steradian."*

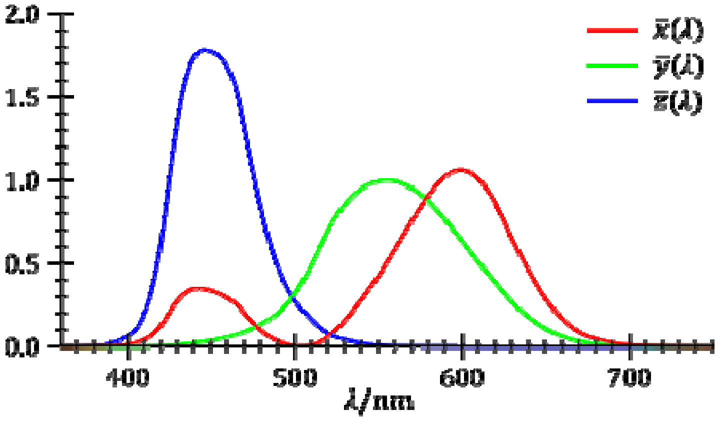


Figure 2-5 The CIE standard observer color matching functions, sensitivity plotted against wavelength
Image Credit: Acdx, 2009

However, since lighting simulation is concerned with light sources are of varying wavelengths (or colors) beyond 540nm (green) monochromatic, and human vision is less sensitive at other wavelengths, a much lower efficacy is typically used. Empirically, RADIANCE uses an efficacy of 179 lumens per watt, representative of the efficacy of white light (Ward, The RADIANCE Lighting Simulation and Rendering System, 1994). To account for the same effect (sensitivity of human vision at various wavelengths), RADIANCE uses a normalized ratio of 0.265: 0.670: 0.065 (sum to 1) for the three primaries, red, green, and blue, when converting from the trichromatic radiometric results to photometric luminance values.

$$Luminance\ (lumens/steradian/sq.meter) = 179\ \times\ (0.265 \times R\ +\ 0.670 \times G\ +\ 0.065 \times B)$$

$$where\ R, G\ and\ B\ are\ trichromatic\ radiance\ values$$

Eq. 2-6

The same ratios are implemented in the new tool to account for the human vision system.

2.3. Light Scattering

Local Illumination models describe how light interacts with objects in a scene, describing the effects of scattering (reflection, or transmission) and absorption. Considering the context of architecture lighting, we can make assumptions to limit the effects to be modeled, excluding phenomena such as fluorescence and phosphorescence. Further assumptions such as a vacuum-like medium within which light travels, and *Helmholtz reciprocity* allows the models to be concise and manageable.

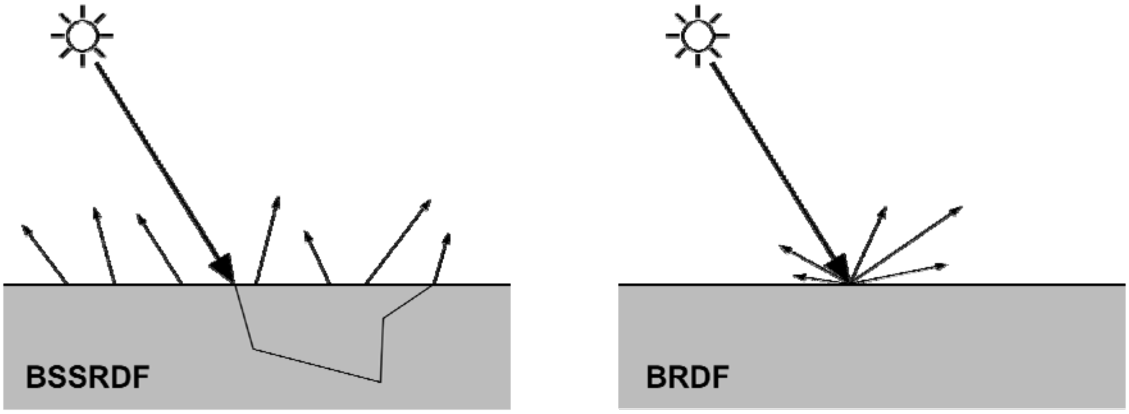


Figure 2-6 Light scattering within a material before exiting in the BSSRDF model (left), Incident and reflected light at the same surface location in the BRDF model (right) (Jensen, 2001)

2.3.1. The BSSRDF

The *Bidirectional Scattering Surface Reflectance Distribution Function* (BSSRDF) (Nicodemus, Richmond, Hsia, Ginsberg, & Limperis, 1977) is a general model of light scattering. Light incident on a non-metallic surface is often scattered within the material before exiting the surface at different locations, angles and intensities (*subsurface scattering*, Figure 2-6). The BSSRDF, S, relates the differential outgoing radiance L_r at position x.in the direction $\vec{\omega}$ to the differential incident flux $d\Phi_i$ at another location x' from some direction $\vec{\omega}'$.

$$S(x,\vec{\omega},x',\vec{\omega}') = \frac{dL_r(x,\vec{\omega})}{d\Phi_i(x',\vec{\omega}')}$$

Eq. 2-7

While this function accurately captures the effects of light scattering, it is costly to model and evaluate since the consideration of every point *x* within the scene involves multiple calculations of S, for large sets of x' and corresponding samples of $\vec{\omega}'$ over all visible directions at each x'. Fortunately, subsurface scattering is of concern more as a visual effect than performance consideration in building design. This allows the simplification and assumption that reflected light leaves a surface at the same location at which it was incident (BRDF).

2.3.2. The BRDF

The *Bidirectional Reflectance Distribution Function* (BRDF), introduced by the same authors, is an approximation of the BSSRDF where incident flux and reflected radiance are assumed to occur at the same surface location (Figure 2-6). The BRDF function f_r, relates the differential outgoing radiance L_r at position x.in the direction $\vec{\omega}$ to the differential irradiance $E_i$at the same location from some direction $\vec{\omega}'$.

$$f_r(x,\vec{\omega}',\vec{\omega}) = \frac{dL_r(x,\vec{\omega})}{dE_i(x,\vec{\omega}')} = \frac{dL_r(x,\vec{\omega})}{L_i(x,\vec{\omega})(\vec{\omega}'\cdot\vec{n})d\vec{\omega}'}$$

Eq. 2-8

The BRDF follows *Helmholtz reciprocity*; it is independent of the incident and exitant directions.

$$f_r(x,\vec{\omega}',\vec{\omega}) = f_r(x,\vec{\omega},\vec{\omega}')$$

Eq. 2-9

Following energy conservation, a surface cannot reflect more light than it receives. Thus:

$$\int_{\Omega} f_r(x,\vec{\omega}',\vec{\omega})(\vec{\omega}'\cdot\vec{n})d\vec{\omega}' \leq 1 \quad \forall\vec{\omega}$$

Eq. 2-10

The BRDF for ideal diffuse, ideal specular and transparent materials are well established. Most materials, however, are a combination of diffuse and glossy specular.

2.3.3. Ideal Diffuse Reflection

Lambertian or *Ideal Diffuse Reflection* assumes that the reflection of incident light is perfectly random, i.e. the reflected radiance $L_r(x,\vec{\omega})$ is constant in all directions (Figure 2-7). This reduces the BRDF to $f_{r,d}(x)$ since it is regardless of direction.

$$L_r(x,\vec{\omega}) = f_{r,d}(x)\int_{\Omega} dE_i(x,\vec{\omega}) = f_{r,d}(x)E_i(x)$$

Eq. 2-11

Since $\int\limits_{\Omega}(\vec{n}\cdot\vec{\omega})d\vec{\omega} = \pi$ and diffuse reflectance ρ_d is defined as the ratio of reflected to incident flux,

$$\rho_d(x) = \frac{d\Phi_r(x)}{d\Phi_i(x)} = \frac{L_r(x)dA\int\limits_{\Omega}(\vec{n}\cdot\vec{\omega})d\vec{\omega}}{E_i(x)dA} = \pi f_{r,d}(x)$$

$$\Rightarrow f_{r,d}(x) = \frac{\rho_d}{\pi}$$

Eq. 2-12

which express the BRDF of Lambertian surfaces as simply 1/π that of the reflectance.

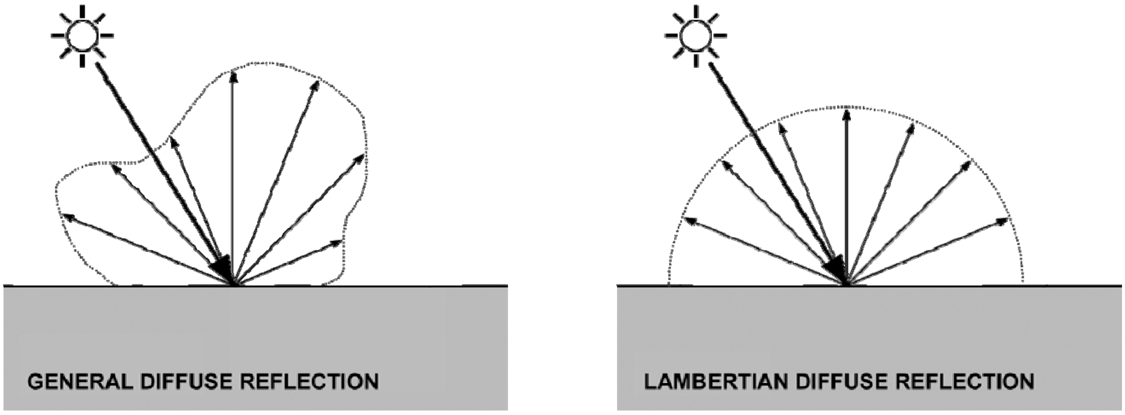


Figure 2-7 General diffuse reflection and Lambertian diffuse reflection

2.3.4. Ideal Specular Reflection

Specular Reflection assumes that a smooth surface reflects incident light in a specific direction. While most smooth surfaces have imperfections resulting in some scattering of the reflected light, this phenomenon is termed glossy reflection. The new tool uses an empirical Gaussian model (section below) to model glossy reflection. Perfect specular reflection assumes the surfaces to be perfectly smooth with a single mirror direction for the reflected light (Figure 2-8).

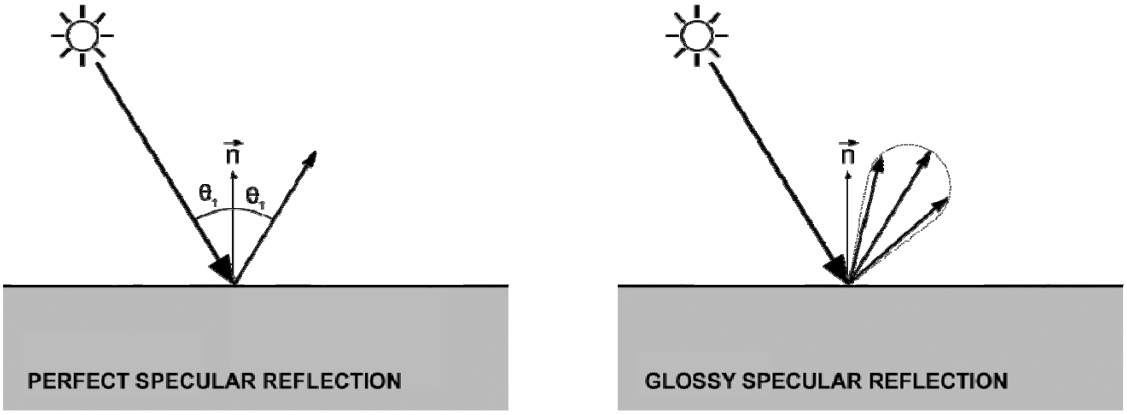


Figure 2-8 Perfect specular (mirror-like) reflection (left) and glossy reflection (right)

The reflected radiance, in the reflected (mirrored) out-going direction, due to specular reflection is thus only affected by the specular reflectance ρ_s of the material:

$$L_r(x, \vec{\omega}_s) = \rho_s(x) L_i(x, \vec{\omega}')$$

Eq. 2-13

Where the reflected (mirrored) direction is:

$$\vec{\omega}_s = 2(\vec{\omega}' \cdot \vec{n})\vec{n} - \vec{\omega}'$$

Eq. 2-14

The BRDF of ideal specular materials is thus simply:

$$f_{r,s}(x, \vec{\omega}, \vec{\omega}') = \begin{cases} \rho_s \mid \vec{\omega} = \vec{\omega}_s \\ 0 \mid \vec{\omega} \neq \vec{\omega}_s \end{cases}$$

Eq. 2-15

2.3.5. *Transparent Materials*

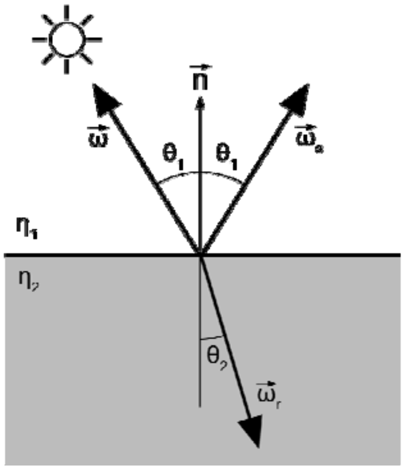


Figure 2-9 Geometry of perfect specular reflection and refraction

When light moves from a medium of a given refractive index η_1 into a second medium with refractive index η_2, both reflection and transmission (in refracted direction) of the light may occur. The reflected direction is already described in Eq. 5-14, and the refracted direction $\overrightarrow{\omega_r}$ is computed from Snell's law, which relates the angle of incidence $\theta_1$to the angle of refraction θ_2.

$$\eta_1 \sin\theta_1 = \eta_2 \sin\theta_2$$

Eq. 2-16

$$\overrightarrow{\omega_r} = -\frac{\eta_1}{\eta_2}(\vec{\omega} - (\vec{\omega}\cdot\vec{n})\vec{n}) - \left(\sqrt{1 - \left(\frac{\eta_1}{\eta_2}\right)^2 (1 - (\vec{\omega}\cdot\vec{n})^2)}\right)\vec{n}$$

Eq. 2-17

Fresnel Equations gives the amount of reflection and refraction as dependent upon the incident and refracted angles. The coefficients of reflection parallel to plane of incidence $\rho_\parallel$, and perpendicular to plane of incidence $\rho_\perp$ are:

$$\rho_\parallel = \frac{\eta_2 \cos\theta_1 - \eta_1 \cos\theta_2}{\eta_2 \cos\theta_1 + \eta_1 \cos\theta_2}$$
$$\rho_\perp = \frac{\eta_1 \cos\theta_1 - \eta_2 \cos\theta_2}{\eta_1 \cos\theta_1 + \eta_2 \cos\theta_2}$$

Eq. 2-18

For un-polarized light, the *Fresnel Reflection Coefficient* F_r, gives the specular reflectance:

$$F_r(\theta) = \frac{1}{2}({\rho_\parallel}^2 + {\rho_\perp}^2) = \frac{d\Phi_r}{d\Phi_i}$$

Eq. 2-19

Correspondingly, the amount of transmitted light is $1 - F_r(\theta)$.

2.3.6. Opaque Materials (Empirical Gaussian Model)

As described earlier, most opaque materials reflect light in complicated ways. Ward's empirical Gaussian model (Ward, 1992) is physically-based, energy conserving, follows *Helmhotz reciprocity*, and describes materials as a combination of ideal diffuse (Lambertian) and glossy. The degree of glossiness, whether the specular component of a material tends towards ideal (mirror-like) or glossy (reflection in a small cone about the mirror direction), is described by the roughness α of the material. The roughness is the standard deviation (RMS) of the surface slope, following the micro-facet theory of surfaces, which assumes the surface to be composed of many small facets, each with perfect distribution. The assumption is that the average slope angle follows a Gaussian, thus the descriptor for the model.

$$f_r(x, \vec{\omega}, \vec{\omega}') = \frac{\rho_d}{\pi} + \rho_s \frac{e^{-(tan^2\delta/\alpha^2)}}{4\pi\alpha^2\sqrt{(\vec{\omega}' \cdot \vec{n})(\vec{\omega} \cdot \vec{n})}}$$

$$where\ \delta\ is\ the\ angle\ between\ surface\ normal\ \vec{n}\ and\ halfangle\ \vec{h} = (\vec{\omega}+\vec{\omega}')/\|\vec{\omega}+\vec{\omega}'\|$$

Eq. 2-20

The new tool implements this model for all opaque materials. The separation of the diffuse and specular components allows each to be calculated separately, and the separation of the normalization factor $\frac{1}{4\pi\alpha^2\sqrt{(\vec{\omega}'\cdot\vec{n})(\vec{\omega}\cdot\vec{n})}}$ and attenuation factor $e^{-(tan^2\delta/\alpha^2)}$ with lends itself naturally to its use in Photon Mapping; the attenuation factor is used to determine the recursive photon direction when using Russian Roulette to spawn photons, and the normalization factor is included when evaluating the BRDF during final gather.

2.4. The Rendering Equation

Considering only the visible spectrum and steady state conditions, the rendering equation (Kajiya, 1986) describes the radiance leaving a point x, towards the observer (direction $\overrightarrow{\omega}$), as the sum of emitted radiance L_o the reflected incident radiance L_i at that point.

$$L_o(x,\vec{\omega}) = L_e(x,\vec{\omega}) + \int_{\Omega} f_r(x,\vec{\omega},\vec{\omega}')\, L_i(x,\vec{\omega})(\vec{\omega}' \cdot \vec{n})d\vec{\omega}$$

Eq. 2-21

The accuracy of any lighting simulation tool lies in the ability of the tool, via its rendering technique, or algorithm, to solve this equation. By implementing the physically-based BRDFs described above, all parts of the light transport (as necessary for architecture lighting considerations) are considered.

The recursive nature of the rendering equation describing inter-reflections between surfaces in the scene is evident when considering that the incident radiance L_i from a particular direction originated as the emitted radiance $L_o$from another surface point. Eq. 5-21 can then be expressed as:

$$L = L_e + TL$$
$$where\ (Tg)(x,\vec{\omega}) = \int_{\Omega} f_r(x,\vec{\omega},\vec{\omega}')\, g(x,\vec{\omega})(\vec{\omega}' \cdot \vec{n})d\vec{\omega}$$

Eq. 2-22

which expands to:

$$L = L_e + TL_e + T^2L_e + T^3L_e + T^4L_e + \cdots = \sum_{m=0}^{\infty} T^m L_e$$

Eq. 2-23

This implies that an infinite number of reflections have to be considered to solve the rendering equation. However, since the overall radiance contribution of increasing depths in the recursion to the top level emitted radiance becomes geometrically smaller (reflectance is always less than

1 for real materials), the expansion can be reasonably pruned once the contribution falls below a certain threshold.

2.4.1. Path Notation

For ease of describing and classifying light paths, path notation (Heckbert, 1990) is commonly used. Vertices of the light path are designated **L** (light source), **E** (eye), **S** (specular reflection) and **D** (diffuse reflection).

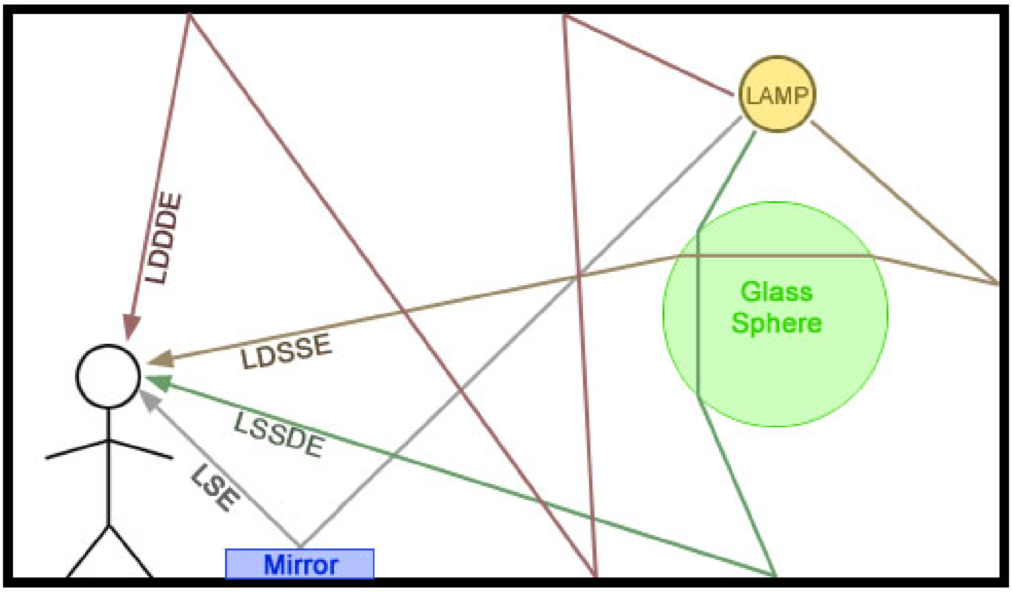


Figure 2-10 Light transport path notation

For abbreviating long paths, regular expressions are used:

- (k)+ denotes one or more k events
- (k)* denotes zero or more k events
- (k)? denotes zero or one k event
- (k|k’) denotes k or k’ event

Caustics, concentration of light caused by specular reflections from curved objects, are represented as LS+ as they reach a surface. If the surface is diffuse and the caustic observed by the view (eye), the entire path is LS+DE. If the surface is specular and the view (eye) observes the caustics via specular reflect, the entire path becomes LS+E

3. Case Studies of Contemporary Practice

Cases studies of lighting simulation within actual design contexts are presented to illustrate the tasks involved and establish quantitative the time and effort baselines for subsequent evaluation of new lighting simulation tool.

3.1. Case Study 1 – Lighting Simulation in Integrated Concurrent Design

The following case study is presented to exemplify and quantify the issues of time and effort in conducting lighting simulation in integrated design processes. The Center for Building Performance and Diagnostics, in conjunction with United Technologies Research Center, developed integrated solutions for a quick service restaurant (Lam, Loftness, Hartkopf, Huang, Zhai, & Bing, 30 November 2007). To achieve a high performance holistic design solution, the multidisciplinary team collaborated in an integrated concurrent design process. The benefits and challenges of integrated design have been well discussed and documented (Gail, Todd, & Hayter, 2003; Deru & Torcellini, 2004; National Institute of Building Sciences, 2008). Concurrent design attempts to reduce the turn-around time and improve the efficiency of such multidisciplinary effort by conducting the various domain tasks in parallel.

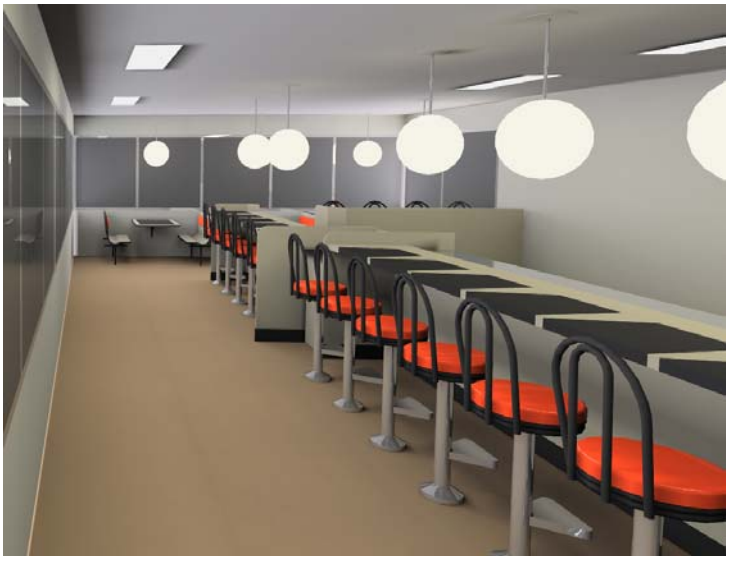


Figure 3-1 Lighting Design of Quick Service Restaurant (Lam, Loftness, Hartkopf, Huang, Zhai, & Bing, 30 November 2007)

Lighting simulation (using RADIANCE) was conducted with to analyze existing day lighting conditions to facilitate optimized lighting layout and control design. From an initial set of design documents including digital 2D CAD drawings and specifications, a series of performance mandates in the various domains were formed. The quantification of desired lighting performance led to the identification of appropriate benchmarks and metrics that can be used to evaluate measure and compare the appropriateness of various design strategies. By decomposing the benchmarks and metrics into fundamental radiance and irradiance levels, lighting simulation tools can be used to compute the latter efficiently.

The objective of lighting simulation was to perform hourly illuminance analysis for 2 Design Days[24] (24 time-steps). However, such results would then require further analysis and processing to become operative information; information that is relevant to and allows design decisions to be made. The initial set of illuminance results are used to define the electric lighting design, and subsequent sets of lighting simulation are used to iteratively evaluate and optimize the design. This remains largely a manual and repetitive task. It is noted that this process of problem formulation and analysis of simulation output requires much user expertise and tacit knowledge. The steps involved are:

I. Preparation of Simulation Input Files (RADIANCE)
II. Preparation of Simulation Parameters
III. Simulation run (RADIANCE)
IV. Results post-processing
V. Repeat process for different lamp layouts and controls
    i. Energy impact evaluated by concurrent team (need to pass information to team)
    ii. Design by multidisciplinary team

[24] A Design Day is a typical day with representative values (daylight and sunlight conditions) selected for simulation to avoid large numbers of simulations. Since design is concerned with *typical* conditions, the slight variations between days available from hourly metrological data files offer resolution beyond what is necessary for lighting design. It is standard practice (as well as in industry standards such as LEED), to use conditions in the 2 equinox days (usually Mar and Sept 21st, depending on calendar year) as Design Days for lighting simulation and analysis. In this case, there were approximately 12 day-lit hours per equinox day, resulting in the need for 24 hourly lighting simulations.

Simulation Input Files	Geometry	*Convert solid model to surface-based description* *Vertex reordering of surfaces with openings*
	Material Information	*Convert material types to RADIANCE material types* *Covert material properties to RADIANCE syntax & units of measure* *Process openings as secondary light sources*
	Lamp Properties	*Convert lamp specifications to RADIANCE syntax & units of measure* *Process reflectors as secondary light sources*
	Sky Definition	*Define RADIANCE sky luminance function* *Define ground plane and sky hemisphere*
Simulation Parameters	2 Camera Views	*Define scene views* *Define camera types*
	Batch Files	*Define simulation parameters* *(model extents & variability, exposure, recursion depth, resolution)* *Generate ordered and linked sets of input files per time-step* *Generate batch command files*

Table 3-1 Breakdown of tasks to obtain first successful simulation run

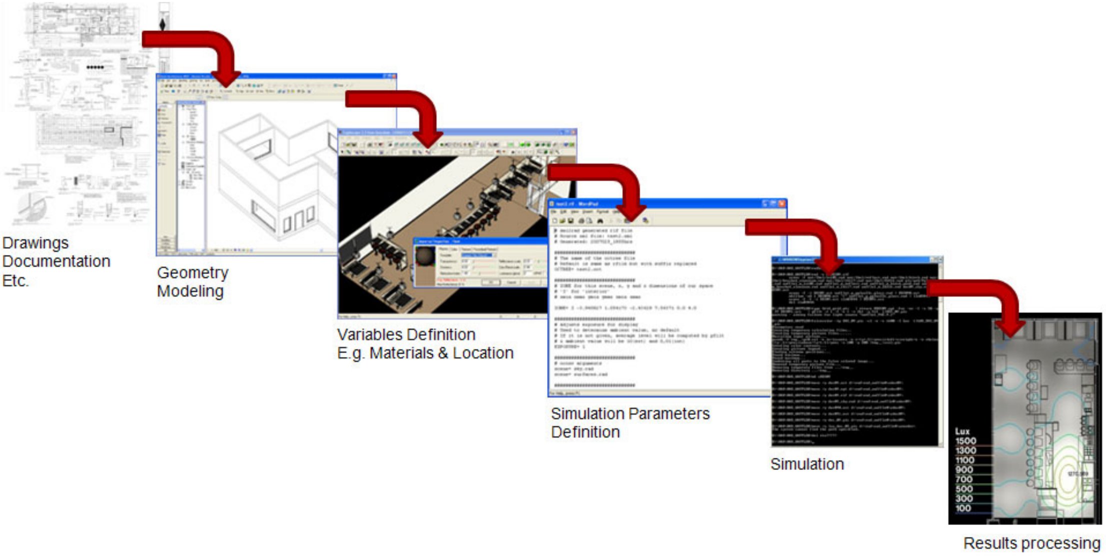


Figure 3-2 Multiple tools and duplicate entry of data

Even when the objectives for lighting simulation are well-defined, there remains much manual work that requires user expertise. The design information, essentially a building model, has to be remodeled with appropriate semantics for lighting simulation. Additional assumptions, such as geometric abstraction and material properties, have to be made. Often, part of this information is related to or duplicated by concurrent work in other domains. The sharing and checking of such information is done manually, error-prone and time consuming. Following the completion of the remodeling, much expertise is required to conduct the simulation and process the results. The entire process is thus time consuming, involves multiple tools and the problem of redundant data-entry and information exchange is further compounded by the non-interoperability of these tools (Figure 3-2). The time associated with each step of the simulation workflow is tabulated below (Table 3-2). The main user time and effort burden is the manual preparation of simulation input files and simulation parameters, taking some 9.6 hours. While the simulation run takes 1 hour per time-step, this is computation time and does not require any user effort. Additionally, since each time-step simulation is independent, they can be run concurrently on several computers to reduce the actual time the user has to wait for results. At the end of the simulation run, at took approximately 3 hours to post-process the results into operative information in the form of 12-sets of false-color images to visualize the illuminance distribution.

In each subsequent simulation iteration (red-text in Table 3-2), the new lamp specifications, geometry, and layout, have to be updated in the simulation input files. This takes approximately 2 hours. Thereafter, the same simulation run and post-processing tasks (highlighted in brown) have to be conducted to generate the necessary operative information.

		Manual Process
		• *All information in CAD drawings and spreadsheets* • *Expert User*
Simulation Input Files	Geometry	7 Hours
	Material Information	1 Hour
	Lamp Properties	0.5 Hour
	Sky Definition	0.5 Hour
Simulation Parameters	2 Camera Views	0.1 Hour
	Batch Files	0.5 Hour
Simulation Run	RADIANCE Computation time	24 Hours (1 hour each time-step)
Post processing	Generate false-color images from results	3 Hours
Iterative Run	Update Geometry and Lamps in Input Files Generate new batch file	2 Hours
Simulation Run	RADIANCE Computation time	24 Hours (1 hour each time-step)
Post processing	Generate false-color images from results	3 Hours
Iterative Run	…	

Table 3-2 Time taken for each step of the simulation workflow

Additional impediments to the efficient use of simulation tools in concurrent deign arise from the progressive nature of design development. First, the varying levels of detail (LOD) a design solution possesses as it develops and evolves limit the consistent use of lighting simulation tools and metrics through the stages of design. Since different facets of information are only available or refined at different stages of design, this entails the inevitable use of different performance metrics and corresponding tools at the various stages depending on what information is available. This hinders effective performance comparisons and progress tracking central to implementing integrated processes.

Second, some parameters in each domain task may be dependent on other domains or that information necessary for the tasks may not yet be available at that stage of design. Assumptions made in each domain may be conflicting with related assumptions in other domains, but the level of detail of the design may not yet be high enough for effective conflict resolutions, leading to consequences downstream.

In keeping with the observations from the literature review, lighting simulation tools have shown to be potentially useful in supporting integrated concurrent design, though currently falling short by being too time-consuming and requiring significant expertise and resources to use. The varying LODs a design undergoes throughout the design process also pose significant challenges to effective simulation tool usage. The pertinent points within these two issues are summarized as follows:

Too time consuming

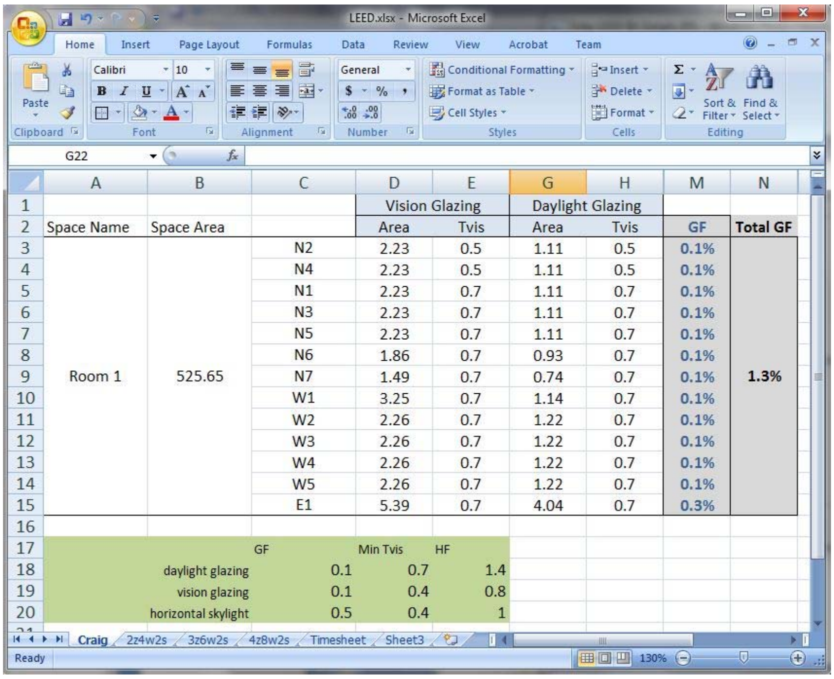
- Semantic differences – redundant effort in remodeling
- Interoperability – difficulty in information exchange between tools
- Operative information – repetitive and difficulty in processing results

Varying LOD

- Consistent metrics – difficulty in implementing consistent technical method
- Information availability – missing information and difficulty in error-checking

3.2. Case Study 2 – Performance Benchmarking

The following case study is presented to illustrate the workflow in lighting performance benchmarking to aid performance-based building design, and to similarly establish and quantify user time and effort costs in acquiring operative information for design. The Center for Building Performance and Diagnostics (CBPD) performed lighting performance benchmarking for a university building retrofit project to help the design team synthesize a green building design. The LEED rating system (U.S. Green Building Council, 2005) was selected by the design team; LEED Credits EQ 8.1 and EQ 8.2 calculation[25] results, in stipulated tabulation formats, were identified as the required operative information.



	A	B	C	D	E	G	H	M	N
1				Vision Glazing		Daylight Glazing			
2	Space Name	Space Area		Area	Tvis	Area	Tvis	GF	Total GF
3	Room 1	525.65	N2	2.23	0.5	1.11	0.5	0.1%	1.3%
4			N4	2.23	0.5	1.11	0.5	0.1%	
5			N1	2.23	0.7	1.11	0.7	0.1%	
6			N3	2.23	0.7	1.11	0.7	0.1%	
7			N5	2.23	0.7	1.11	0.7	0.1%	
8			N6	1.86	0.7	0.93	0.7	0.1%	
9			N7	1.49	0.7	0.74	0.7	0.1%	
10			W1	3.25	0.7	1.14	0.7	0.1%	
11			W2	2.26	0.7	1.22	0.7	0.1%	
12			W3	2.26	0.7	1.22	0.7	0.1%	
13			W4	2.26	0.7	1.22	0.7	0.1%	
14			W5	2.26	0.7	1.22	0.7	0.1%	
15			E1	5.39	0.7	4.04	0.7	0.3%	
16									
17			GF	Min Tvis	HF				
18		daylight glazing	0.1	0.7	1.4				
19		vision glazing	0.1	0.4	0.8				
20		horizontal skylight	0.5	0.4	1				

Figure 3-3 LEED Credit EQ 8.1 Spreadsheet for manual calculation

CBPD calculated LEED Credits EQ 8.1 and EQ 8.2 for the existing building and each proposed design by the design team iteratively. To establish time and effort baselines for subsequent automation work (Chapter 5), both credits were initially manually calculated. The manual calculation procedure assumes well-formed design CAD drawings (plans and sections) and the availability of all necessary material reflectance properties, submittal spreadsheets (Figure 3-3) with calculation formulas, and an expert user familiar with the calculation procedures and

[25] Calculation method and tabulation stipulations covered in detail in Chapter 5.1.

spreadsheets, without making any mistakes. The building is a single open plan office with 13 windows (6 unique types). The times taken to perform manual calculation of both LEED credits are as follows:

Model Description	LEED Credit EQ 8.1	LEED Credit EQ 8.2
1 Zone 13 Windows (6 unique types)	**5 minutes**	**10 minutes**

Table 3-3 Time taken to perform manual LEED credits calculations

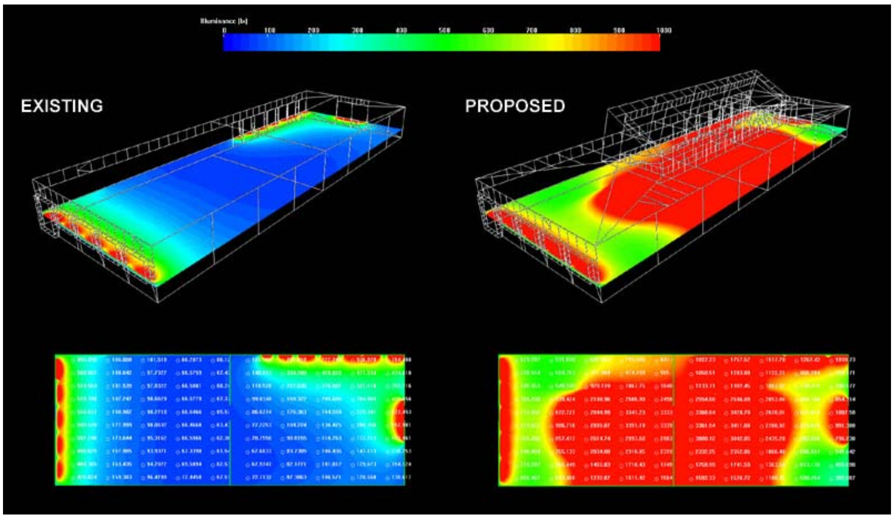


Figure 3-4 Lighting Simulation for LEED EQ 8.1 and illuminance distribution analysis

To provide better accuracy in LEED EQ 8.1 calculation[26], and to provide visualization of the daylight illuminance distribution in the space for better understanding of the lighting performance, CBPD performed lighting simulation of the space in RADIANCE. The simulation workflow is similar to that in Case Study 1 earlier.

I. Preparation of (RADIANCE) Simulation Files
II. Simulation run (RADIANCE) 24 time-steps
III. Results post-processing
IV. Iterate for new design

The time taken for each stage of the workflow is tabulated in Table 3-4 below.

[26] LEED EQ 8.1 can also be calculated via lighting simulation. The credit is awarded if a minimum daylight illumination of 25 footcandles (269 lux) on a 30” (762 mm) work-plane in 75% of regularly occupied areas. The simulation must be under clear sky conditions at noon on the equinox, with a 2’ (610 mm) calculation grid.

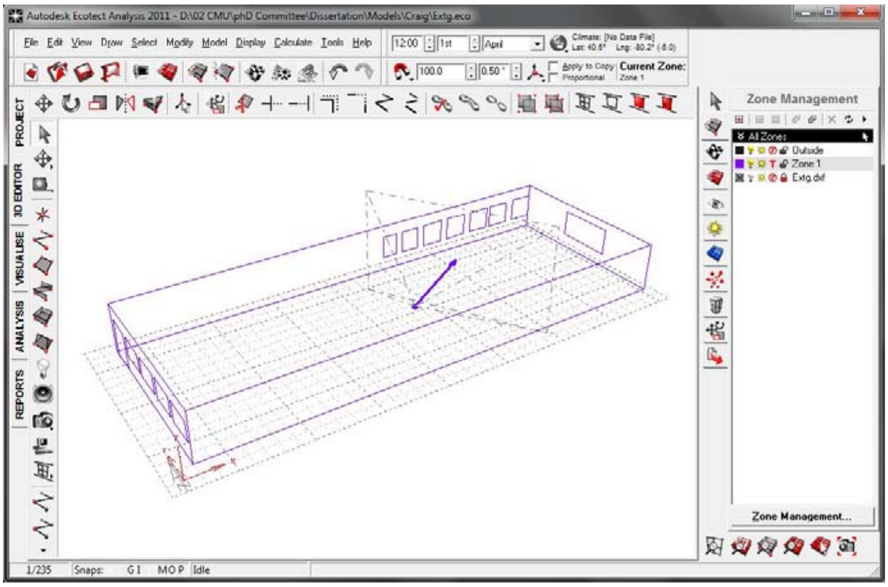


Figure 3-5 Using Ecotect to prepare RADIANCE simulation input files

To illustrate the benefits of partial interoperability offered by middleware as discussed earlier (Chapter 1.2.4) the lighting simulation model (Step I above) was also prepared using Ecotect (Autodesk, 2009). Given a well-formed 3-D AutoCad (*.dwg) design model, Ecotect imports the geometric information within a graphical user interface for quick remodeling and definition of simulation parameters. Ecotect is then able to export the information as RADIANCE simulation input files. The rest of the simulation workflow (Steps II – IV) is the same.

STEP I – Preparation of (RADIANCE) Simulation Files using Ecotect

- Import design geometry, redraw in surfaces
- Materials, location, and camera definition
- Export to RADIANCE
- Manually correct RADIANCE input files
- Manually prepare batch input files

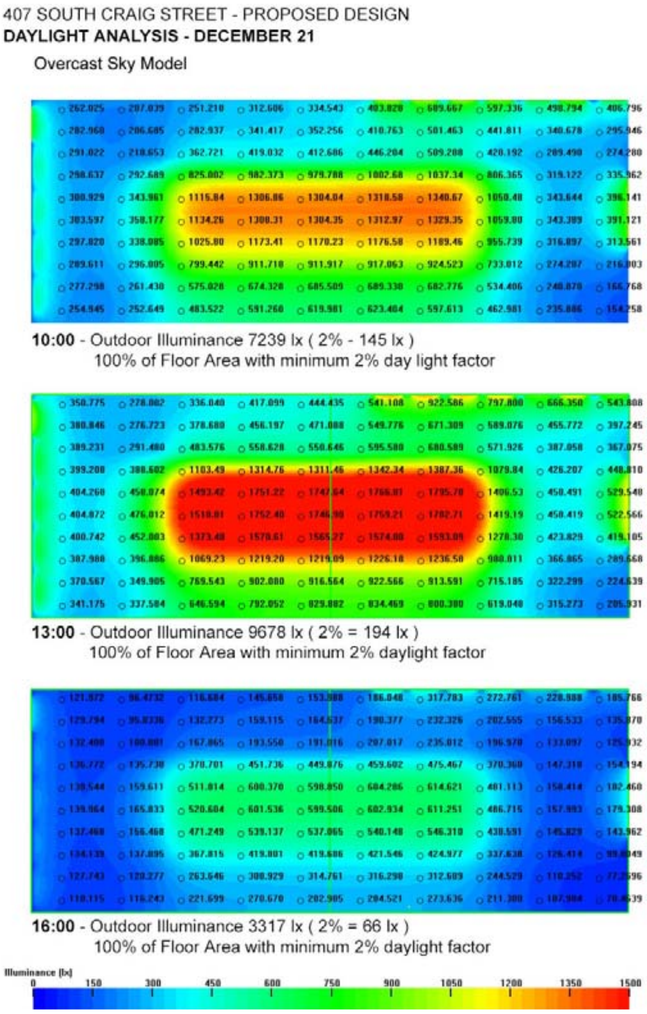
The time taken to prepare simulation files is reduced from 6 hours to 10 minutes (Table 3-4) since graphical input of geometric information is much easier than manual input of numerical values such as coordinate-points information. Likewise, the Ecotect GUI makes it easier to define and inspect material properties, sky and camera definitions, and simulation parameters (batch files), as compared to text-based definitions in the manual process.

In the iterative runs, the geometric and material changes in the design have to be updated in the simulation input files. Using Ecotect reduces the time taken from 1 hour to 10 minutes due to the same ease of graphical input. However, it is noted that this is the same amount of time taken as the initial model preparation, since the time is mostly attributed to navigating the user interface rather than manual keying-in of numerical values.

		Manual Process	Ecotect
		• *All information in CAD drawings and spreadsheets* • *Expert User*	• *Successful CAD drawings import* • *Expert User*
Simulation Input Files Preparation	Geometry	4 Hours	10 Minutes
	Material Properties	0.5 Hour	
	Sky & Camera Definition	0.1 Hour	
	Batch Files	0.5 Hour	
Simulation Run	RADIANCE Computation time	4.5 Hours (15 Minutes each time-step)	
Post processing	Generate false-color images	3 Hours	
Iterative Run	Update Geometry Generate new batch files	1 Hour	10 Minutes
Simulation Run	RADIANCE Computation time	4.5 Hours (15 Minutes each time-step)	
Post processing	Generate false-color images	3 Hours	
Iterative Run	…		

Table 3-4 Time taken for each step of the simulation workflow

In both manual and Ecotect-assisted lighting simulation, the post-processing stage entails the analysis of simulation results and preparation of illuminance false-color visualizations (Figure 3-6). In addition, perspective luminance renders were also generated to better visualize the lighting design, and compare performance between design iterations (Figure 3-7). The post-processing entails the manual use of RADIANCE utilities, amounting to some 3 hours.



407 SOUTH CRAIG STREET - PROPOSED DESIGN

DAYLIGHT ANALYSIS - DECEMBER 21

Overcast Sky Model

262.025	207.039	251.210	312.606	334.543	403.828	609.667	597.336	498.794	406.796
282.960	206.685	282.937	341.417	352.256	410.763	501.463	441.811	340.678	295.946
291.022	218.653	362.721	419.032	412.606	446.204	509.208	420.192	289.490	274.280
298.637	292.689	825.002	982.373	979.788	1002.68	1037.34	806.365	319.122	335.962
300.929	343.961	1115.84	1306.86	1304.04	1318.58	1340.67	1050.48	343.644	396.141
303.597	358.177	1134.26	1308.31	1304.35	1312.97	1329.35	1059.80	343.389	391.121
297.020	338.085	1025.80	1173.41	1170.23	1176.58	1189.46	955.739	316.897	313.561
289.611	296.005	799.442	911.710	911.917	917.063	924.523	733.012	274.207	216.803
277.298	261.430	575.028	674.328	685.509	689.330	682.776	534.406	240.070	166.768
254.945	252.649	483.522	591.260	619.981	623.404	597.613	462.981	235.886	154.258

10:00 - Outdoor Illuminance 7239 lx (2% - 145 lx)
100% of Floor Area with minimum 2% day light factor

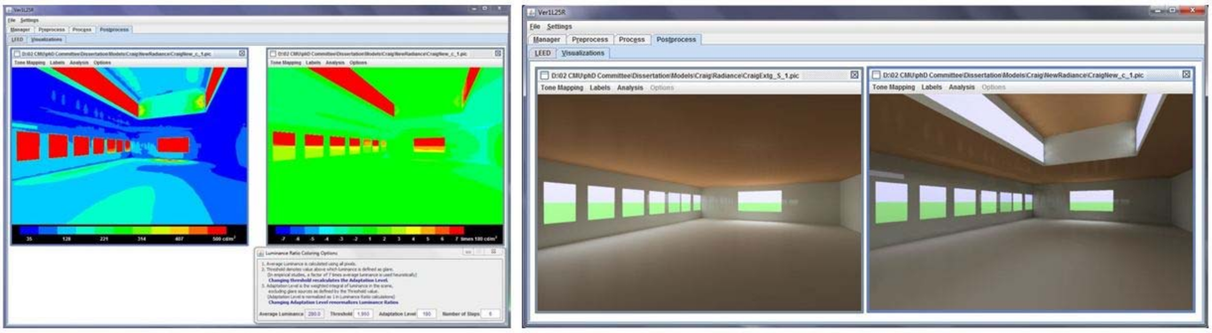
350.775	278.002	336.040	417.099	444.435	541.108	922.586	797.800	666.350	543.808
380.846	276.723	378.680	456.197	471.088	549.776	671.309	589.076	455.772	397.245
389.231	291.480	483.576	558.628	550.646	595.580	680.589	571.926	387.058	367.075
399.208	388.602	1103.49	1314.76	1311.46	1342.34	1387.36	1079.84	426.207	448.810
404.260	450.074	1493.42	1751.22	1747.64	1766.81	1795.78	1406.53	450.491	529.548
404.872	476.012	1518.01	1752.40	1746.90	1759.21	1782.71	1419.19	458.419	522.566
400.742	452.003	1373.48	1570.61	1565.27	1574.00	1593.09	1278.30	423.829	419.105
387.988	396.886	1069.23	1219.20	1219.09	1226.18	1236.58	980.011	366.865	289.668
370.567	349.905	769.543	902.080	916.564	922.566	913.591	715.185	322.299	224.639
341.175	337.584	646.594	792.052	829.882	834.469	800.380	619.048	315.273	205.931

13:00 - Outdoor Illuminance 9678 lx (2% = 194 lx)
100% of Floor Area with minimum 2% daylight factor

121.872	96.4732	116.684	145.658	153.988	186.048	317.783	272.761	228.988	185.766
129.794	95.8336	132.273	159.115	164.637	190.377	232.326	202.555	156.533	135.870
132.400	100.801	167.865	193.550	191.016	207.017	235.012	196.970	133.097	125.932
136.772	135.730	370.701	451.736	449.876	459.602	475.467	370.360	147.318	154.194
138.544	159.611	511.814	600.370	598.850	604.286	614.621	401.113	158.414	182.460
139.964	165.833	520.604	601.536	599.506	602.934	611.251	486.715	157.993	179.308
137.460	156.468	471.249	539.137	537.065	540.148	546.310	438.591	145.829	143.962
134.139	137.895	367.815	419.801	419.686	421.546	424.977	337.638	126.414	99.8049
127.743	120.277	263.646	308.929	314.761	316.298	312.609	244.529	110.252	77.2596
118.115	116.243	221.699	270.670	282.905	284.521	273.636	211.380	107.904	78.4639

16:00 - Outdoor Illuminance 3317 lx (2% = 66 lx)
100% of Floor Area with minimum 2% daylight factor

Figure 3-6 Illuminance false-color visualizations



Ver1L25R
File Settings
Manager Preprocess Process Postprocess
LEED Visualizations
D:\02 CMU\phD Committee\Dissertation\Models\Craig\Radiance\CraigExtg_S_1.pic
D:\02 CMU\phD Committee\Dissertation\Models\Craig\NewRadiance\CraigNew_c_1.pic
Tone Mapping Labels Analysis Options
Luminance Ratio Coloring Options

False Color and Luminance Ratio Analysis

Comparing results between design versions

Figure 3-7 Comparative analyzes of design iterations

4. Implementation of Lighting Simulation Engine

(Jensen) Photon Mapping is selected as suitable for simulating high performance building designs, and implemented with slight modifications with consideration of typical building lighting design contexts.

4.1. RADIANCE - Monte Carlo (Backwards) Ray Tracing

As mentioned in literature review, RADIANCE is the most widely available and used lighting simulation tool in building design practice. While it has been widely validated, a detailed review below of its rendering techniques and limitations reveals that high performance building designs that are increasingly becoming commonplace involve highly-reflected light paths that RADIANCE is not suited to handle.

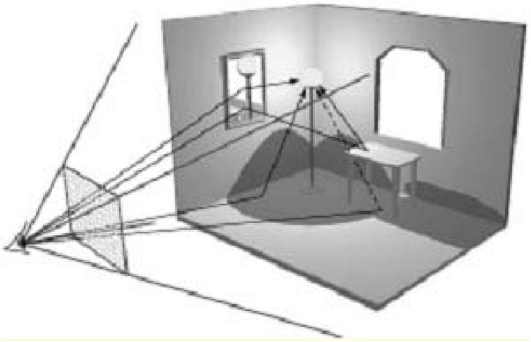


Figure 4-1 Backwards Ray Tracing (Autodesk, 1999)

4.1.1. Implementation in RADIANCE

The fundamental technique used by RADIANCE (Larson & Shakespeare, 2003) is backwards ray tracing that traces ray paths, originating from the camera (eye) position, through pixels in an imaginary plane (the bitmap to be generated), into the scene model. At the first intersection the primary ray encounters with an object in the scene, the radiance towards the observer can be calculated based on the surface BRDF. Depending on the material of this surface, the radiance is calculated by casting three additional types of rays; shadow, reflected and refracted rays. This solves the rendering equation by expressing the incident radiance L_i as two separate terms:

$$L_i = L_{i,l} + L_{i,d|s}$$

Eq. 4-1

Where $L_{i,l}$is incident radiance due to direct illumination from light sources, paths **L(S|D)E**; **(S|D)** depending on the type of material at the primary ray surface intersection, and $L_{i,d|s}$ is from indirect illumination where light from sources have been at least diffuse or specular reflected at least once; paths **L(S'|D')+(S|D)E**, **(S'|D')** representing the diffuse or specular reflection before being finally reflected towards the observer at the primary ray intersection. In the case where the primary ray intersects a light source, reflectance is assumed as zero; the exitant radiance is simply the emitted radiance of the light source:

$$L_o(x, \vec{\omega}) = L_e(x, \vec{\omega})$$

Eq. 4-2

This covers the conditions where the light paths are **LE**. Together with use of the three types of rays described above, this technique covers all possible light paths **L(L|D)*E**.

Shadow rays are cast from the surface intersection point towards light sources to determine if there is direct illumination $L_{i,l}$ at the primary ray intersection (no surface intersections before the shadow ray reaches the light source). (Specular) reflected and refracted rays are cast in all directions$\vec{\omega}'$.over the hemisphere at the primary ray intersection point to evaluate the irradiance from surface inter-reflections $L_{i,d|s}$. At subsequent reflected or refracted ray surface intersections, additional shadow, reflected and refracted rays are cast in a recursive manner. This represents the recursive evaluation of the second and higher order T^m in Eq. 2-23, thus solving the entire rendering equation. To limit the expansion, the recursion is halted when one of the following conditions occurs:

1. The intersected surface is a light source (for which reflectance is approximated as zero)
2. The ray has reflected more than a specified number of times (default limit is 6)
3. The product of all previous reflectance is below a specified values (default limit is 0.005)

Additionally, the number of reflected and refracted rays cast from each point is determined stochastically. Instead of sending excessively large numbers of specular rays (samples) to achieve accurate $L_{i,d|s}$, uniformly weighted Monte Carlo sampling is used to choose directions about the mirror and transmitted directions.

To reduce the number of samples further, irradiance caching and interpolation is used. By assuming that indirect illumination tends to change slowly (low-frequency) over the hemisphere, a lower number of samples can be initially taken with larger directional differences, and the irradiance in between the samples interpolated if the irradiance gradient between adjacent samples is below a specified threshold. In the event the threshold is exceeded, additional sub-sampling between the original samples is done to evaluate the irradiance. Since this recursive technique uses a large number of reflected and refracted rays, these specular rays more often than not intersects surface points near locations where irradiance have been previously calculated. The irradiance cache allows previous evaluations to be reused and interpolated, thus reducing the computational expense of the large number of rays. Like the adaptive sub-sampling, the irradiance is not interpolated if adjacent gradients cross a specified threshold.

4.1.2. Limitations

Considering that the technique of RADIANCE fundamentally traces light paths backwards, its limitations are evident. RADIANCE would underestimate incident indirect irradiance when light paths include a high number of reflections between light source and the point of interest (primary ray intersection), such as in predominantly diffuse-lit non-convex spaces, or when light re-directing devices are used. Such design scenarios are increasingly becoming common place in building design as the industry shifts towards high performance, or green, buildings.

Diffuse lighting (indirect ambient light) has been widely accepted as providing more comfortable environments. Figure 4-2 illustrates typical indirect lighting strategies in high performance, or green, buildings. Daylight is reflected via redirecting devices, such as external light shelves in this case, into the interior, towards the ceiling before it is diffusedly reflected towards work surfaces. Likewise, the interior electric lamps points upwards towards the ceiling.

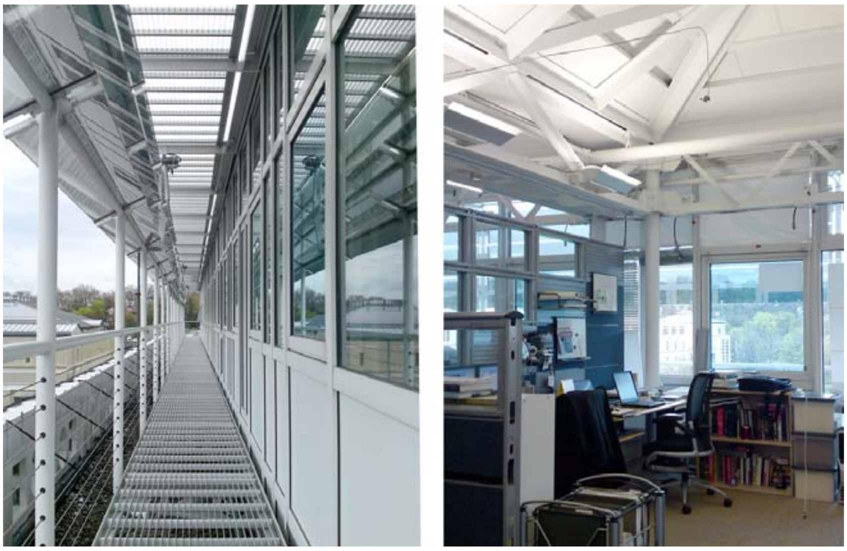


Figure 4-2 Daylight redirection devices (left) and diffuse lighting (right) in the Intelligent Workplace, Center for Building Performance and Diagnostics.

The implication for lighting simulation is that indirect irradiance becomes significant when solving the rendering equation, but as pointed out earlier, excessively large numbers of sample rays would have to be cast recursively to compute this indirect component. Intuitively (Figure 4-3), it is difficult to trace a light path from the observer to the light source when the path had undergone a large number of reflections. Similarly, daylight re-direction devices such as light tubes (Figure 4-4) reflect light paths multiple times before reaching building interiors.

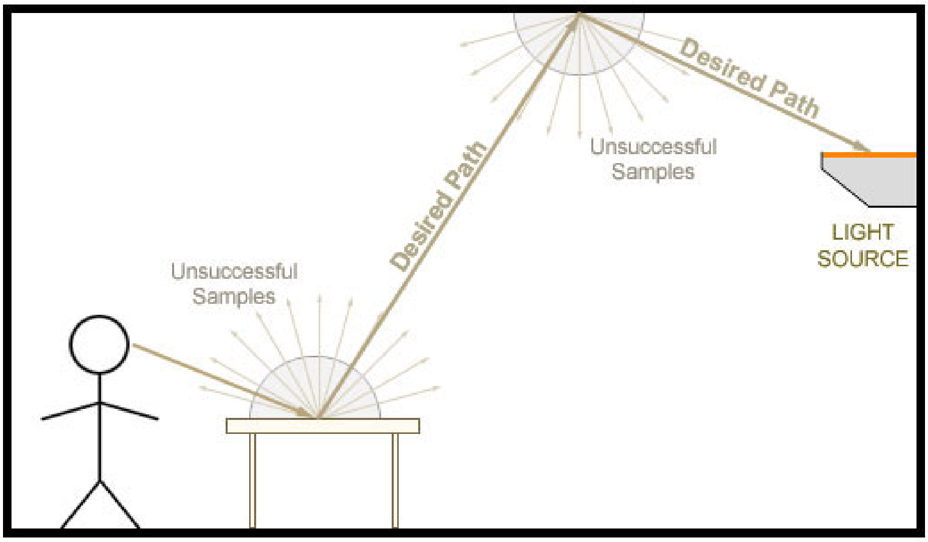


Figure 4-3 Difficulty in tracing light paths back to light source in diffuse lighting scenes using backwards ray tracing

Consequently, backwards ray tracing severely underestimates lighting levels in such cases. Even when larger numbers of samples are taken at each recursive reflected and refracted ray intersection, as made possible by the stochastic method implemented in RADIANCE, it is still insufficient to produce satisfactory results.

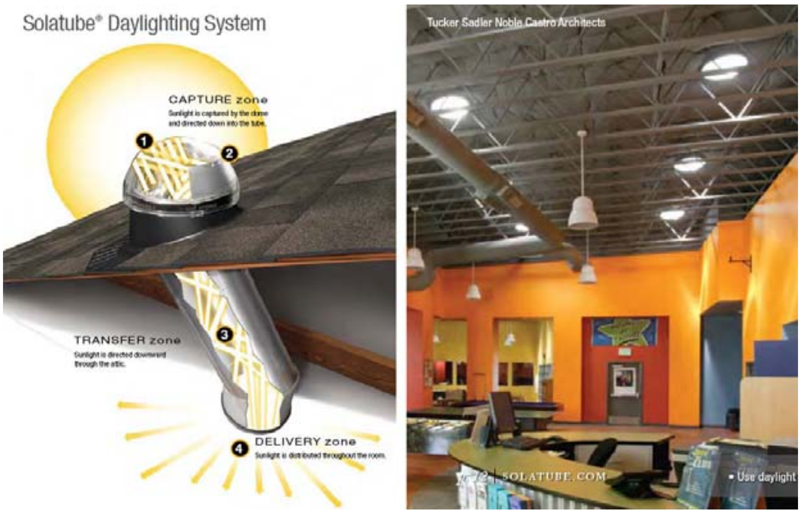

Solatube® Daylighting System
CAPTURE zone
Sunlight is captured by the dome and directed down into the tube.
1
2
TRANSFER zone
Sunlight is directed downward through the attic.
3
4
DELIVERY zone
Sunlight is distributed throughout the room.
Tucker Sadler Noble Castro Architects
SOLATUBE.COM
• Use daylight

Figure 4-4 Example of light tube and typical application
Photo Credit: Solatube International Inc.

As a demonstration, an experimental room (Figure 4-5) lit by a highly reflected 30,000 lumen light source is modeled and simulated using RADIANCE. The light tube is assumed to be perfect specular (actual applications have been shown to be 99.7% specular, or near-ideal specular). A 50% diffuse, 50% specular reflector is hung below the light well, where the mirror director reaches Point A, 3.6m away on a 70% diffuse ceiling.

To estimate the radiance of Point A, the radiant intensity reaching the center of the reflector is approximated by a 2.8m direct path from the light source (indicated by red centerline in Figure 4-5). The radiance towards the mirror direct, and consequently, Point A, is approximated by considering the specular and diffuse reflectance at the reflector separately. Diffuse inter-reflections between all other surfaces are ignored in this estimate.

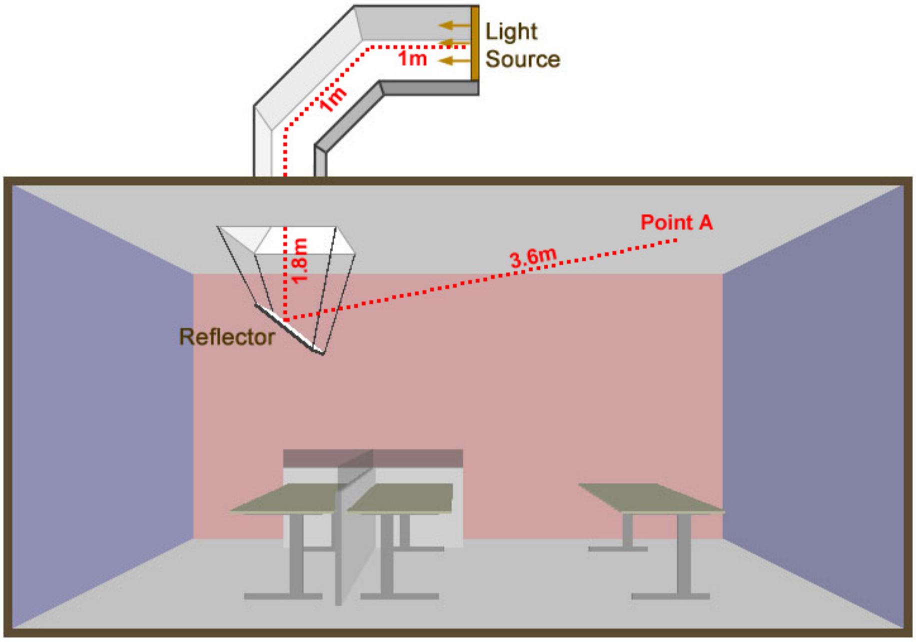


Figure 4-5 Experimental setup of room lit by highly reflected light source

The specular component from the reflector towards Point A can thus be approximated by a 15,000 lumen light source 7.444m away, with incident angle 13°. The radiance due to this component at Point A (towards the observer) is then:

$$
\begin{aligned}
L_s &\approx \frac{E\rho}{\pi} \\
&= \frac{Icos\theta}{r^2} \times \frac{\rho}{\pi} \\
&= \frac{\frac{15{,}000}{\pi} \cdot (cos13°)}{7.4^2} \times \frac{0.7}{\pi} \\
&= 18.9\ cd/m^2
\end{aligned}
$$

Even while this figure considers only a small portion of the irradiance at Point A, the magnitude of the estimate is sufficient to show how Radiance severely underestimates such scenarios. Even at extremely high resolution settings Figure 4-6, the Radiance-calculated value at Point A is only 0.6 cd/m^2, showing how the tool fails to handle such scenarios. Even the distribution, as made more visible by increasing the exposure of the results, shows poor resolution.

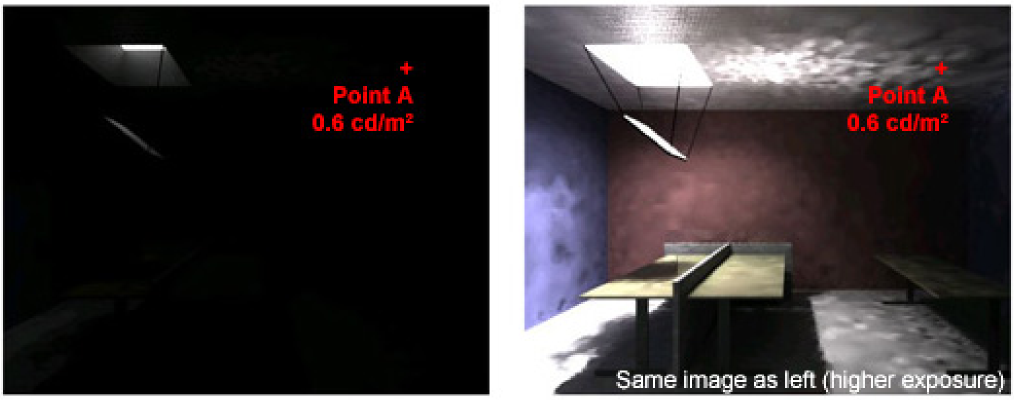


Figure 4-6 RADIANCE simulation of experimental case – severe underestimation and poor distribution

4.2. Classic (Jensen) Photon Mapping

Photon Mapping, as introduced by Jensen (Jensen, 2001) overcomes the issue of highly reflected light scenes by tracing the light paths from the source (forward ray tracing) rather than from the observer towards the light sources. Intuitively, radiant flux from light sources is discretized as packets of photons, which are then traced in the model following the usual local illumination models. At each surface intersection, Russian Roulette, another Monte Carlo technique, is used to stochastically determine the subsequent path of the photon. Photons that intersect diffuse surfaces are stored in a kd-tree, forming the Photon Map (Figure 4-7). As a second (rendering) pass, the usual backwards ray tracing is used for image synthesis, since it is efficient for calculating specular, direct, and diffuse surface to observer light paths (**LS*E**, **L(D|S)E**, and **DS*E** paths). Since each photon represents some radiant flux $\Delta\Phi$, the density of the photon map $\Delta\Phi/\Delta A$ gives the irradiance at that point. Since the BRDF (Eq. 2-8) is a ratio of reflected radiance and irradiance, the reflected radiance $L_r(x, \vec{\omega})$ at some point x, towards some direction $\vec{\omega}$ can be calculated. Photons that have only undergone specular reflections before a last diffuse intersection (**LS+D**) are stored in the caustics photon map. All other photons (**L(S|D)+D**) are stored in the global photon map.

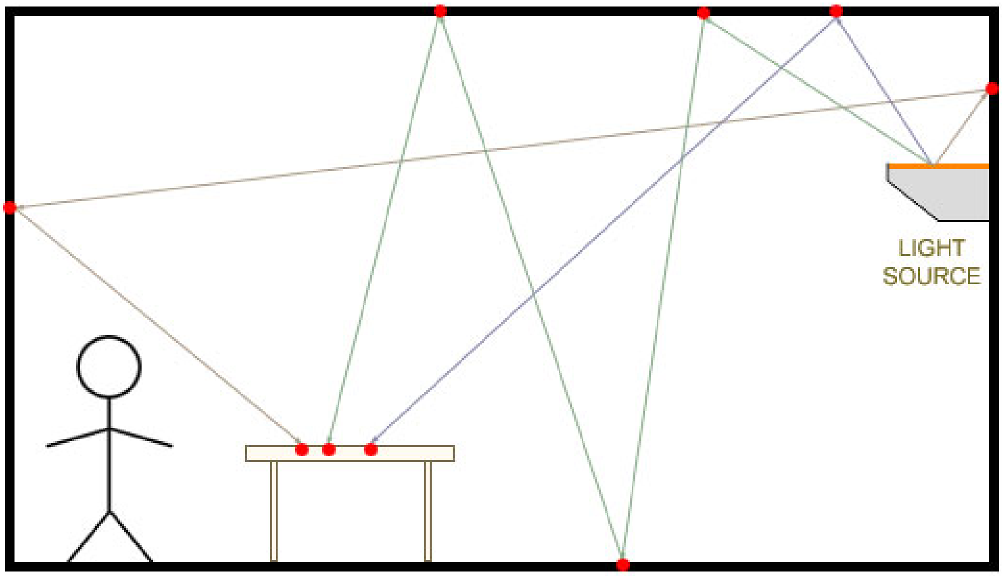


Figure 4-7 Photon Map – Photons that hit a diffuse surface are stored

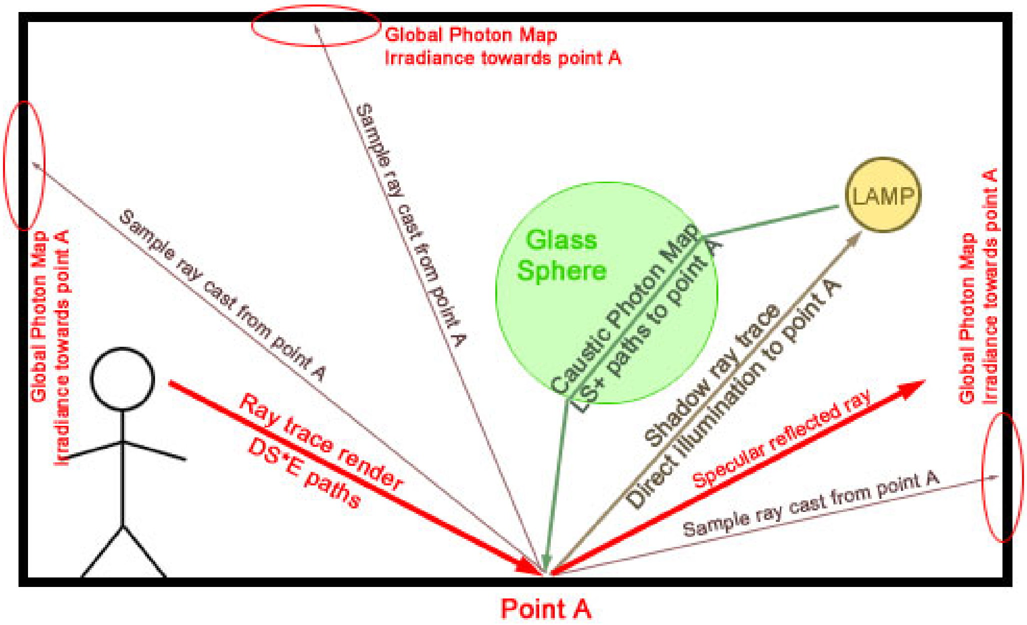

Global Photon Map
Irradiance towards point A
Sample ray cast from point A
Global Photon Map
Irradiance towards point A
Sample ray cast from point A
Glass
Sphere
Caustic Photon Map
LS+ paths to point A
LAMP
Shadow ray trace
Direct illumination to point A
Specular reflected ray
Sample ray cast from point A
Global Photon Map
Irradiance towards point A
Ray trace render
DS*E paths
Point A

Figure 4-8 Radiance of Point A as sum of direct, specular, indirect and caustics components

At each intersection point in the rendering pass (pixel-points of final image) the reflected radiance at that point is expressed as four components; direct, specular, indirect and caustic (Figure 4-8). The reflected radiance term in the rendering equation (Eq. 2-21) can subsequently be expressed as:

$$\begin{aligned} L_r(x,\vec{\omega}) &= \int_{\Omega} f_r(x,\vec{\omega}',\vec{\omega}) L_i(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')\,d\vec{\omega}' \\ &= \int_{\Omega} f_r(x,\vec{\omega}',\vec{\omega}) L_{i,l}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')\,d\vec{\omega}' \\ &\quad + \int_{\Omega} f_{r,S}(x,\vec{\omega}',\vec{\omega})\left(L_{i,c}(x,\vec{\omega}') + L_{i,d}(x,\vec{\omega}')\right)(\vec{n}\cdot\vec{\omega}')\,d\vec{\omega}' \\ &\quad + \int_{\Omega} f_{r,D}(x,\vec{\omega}',\vec{\omega}) L_{i,c}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')\,d\vec{\omega}' \\ &\quad + \int_{\Omega} f_{r,D}(x,\vec{\omega}',\vec{\omega}) L_{i,d}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')\,d\vec{\omega}' \end{aligned}$$

Eq. 4-3

Where the BRDF is split into the diffuse $f_{r,D}$ and specular or glossy $f_{r,S}$ components (not necessarily Lambertian or perfect specular): $f_r(x,\vec{\omega}',\vec{\omega}) = f_{r,S}(x,\vec{\omega}',\vec{\omega}) + f_{r,D}(x,\vec{\omega}',\vec{\omega})$

And $L_{i,l}(x,\vec{\omega}')$ is direct illumination from the light sources, $L_{i,c}(x,\vec{\omega}')$ is caustics – indirect illumination from the light sources via specular reflection or transmission, and $L_{i,d}(x,\vec{\omega}')$ is indirect illumination from light sources that has been reflected diffusely at least once.

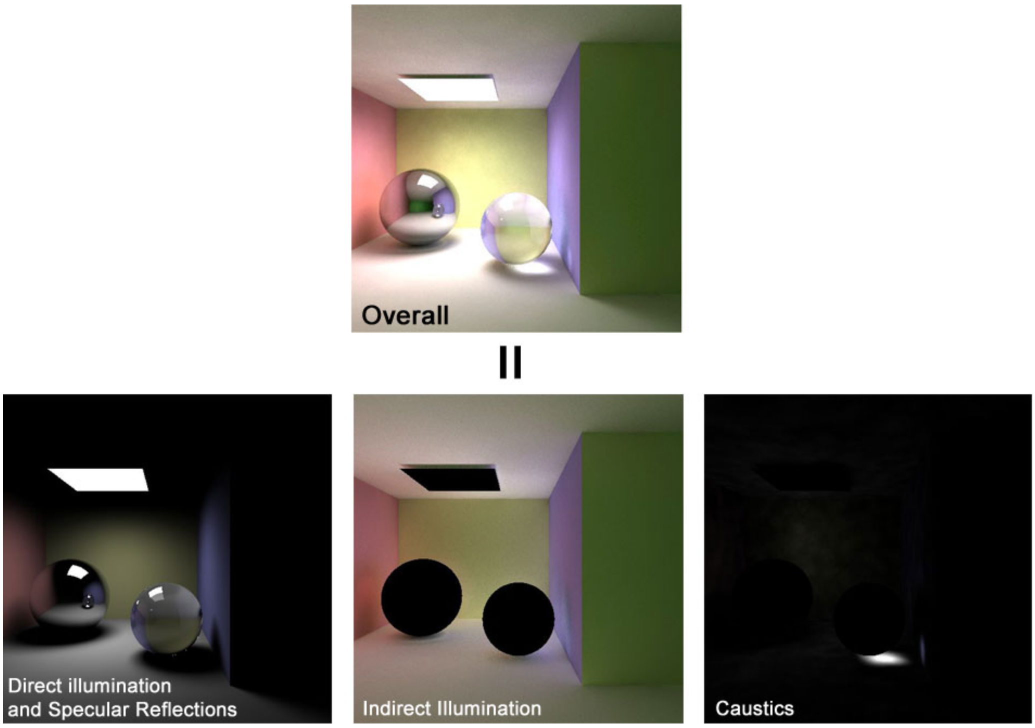


Figure 4-9 4-Component technique in Photon Mapping to solve rendering equation
(Direct illumination and specular reflections solved together in ray trace pass thus presented in single sub-image)

4.2.1. *Implementation*

The ray tracing rendering pass calculates the direct and specular components efficiently. Since caustics photons, by nature of their occurrence, tend to group together and gives good photon density estimates, the radiance due to caustics at the intersection point is calculated using the caustic photon map directly. By contrast, the distribution of the global photon map, while statistically correct, has variance that introduces undesirable visual effects from a photo-realistic imagery point-of-view. Consequently, a number of directions are chosen by importance-sampling the BRDF at the intersection (Point A, Figure 4-8), and sample rays are cast to the next intersection point. The irradiance at Point A is thus the integral of the radiance at each sample point, towards Point A.

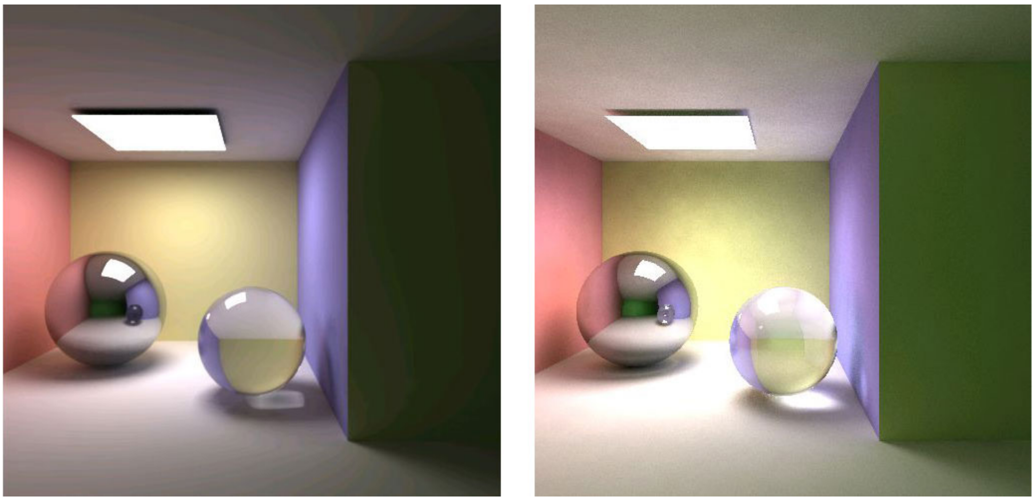


Figure 4-10 RADIANCE (left) and Photon Mapping (right)

Since photon scattering implements physically-based lamp distribution characteristics as well as surface reflection models (using BRDF), the photon map models all light paths comprehensively. As exemplified in Figure 4-10, the **LS+D** paths resulting in the caustics underneath the glass sphere is of poor quality in RADIANCE but visibly correct when using Photon Mapping. The use of Russian Roulette also allows the photon map to efficiently include complex light paths that have high numbers of reflection; instead of scattering each photon in multiple directions (as is necessary in backwards ray tracing), a single reflected photon is emitted by importance sampling the surface BRDF. As an example, if 256 directions are sampled at each reflection (such as necessary in backwards ray tracing), there are more than 4 billion samples after a mere 4 reflections. By contrast, there will only be 4 photons when Russian Roulette is used. To ensure accurate results, a large number of photons (between 0.3 to 4 million in typical architecture scenes) are used. This number is significantly lower than that of multiple sampling, and can be handled by typical workstations reasonably.

In the same figure, the ambient light is brighter when using Photon Mapping since these highly reflected light paths are captured. In the case of the experimental case (Figure 4-5), Photon Mapping is now able to simulate the lighting conditions (Figure 4-11).

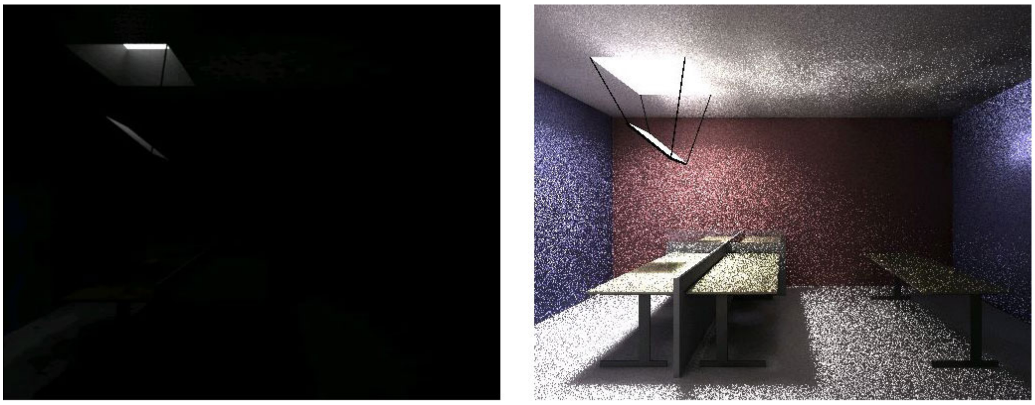


Figure 4-11 Simulation of experimental case. RADIANCE (left), Photon Mapping (right)

4.2.2. Limitations

Since Photon Mapping was developed with a focus on efficient photo-realistic image synthesis, its implementation is focused on visual effects with maximum computational efficiency rather than physically accurate results suitable for lighting design. The global photon map is not used directly but for samplings from the point of interest to reduce visual variance. While a consistent approach would suggest that each sampled point (traced once after the original point of interest) should also be the sum of the four direct, specular, indirect, and caustics components, each calculated consistently as before, approximated values are used instead since the radiance at these sampled points are not observed directly, but only constitute a fractional impact on the original point of interest.

Similarly, the irradiance estimate uses a varying sampling area that gives less variance when the photon density is sparse, and the area used in the estimates are also approximated. Techniques such as approximating areas by convex hulls, cone filtering, and Gaussian filtering, are used to improve the visible quality of the rendered image, while not necessarily improving physically accurate since the focus in this case is efficiency and photo-realism.

4.3. Modified Photon Mapping (New Rendering Technique)

The premise for the Modified Photon Mapping is to use sacrifice approximation techniques adopted in the original technique for efficient photo-realistic image synthesis, and revert to the original calculation basis so that the technique is focused on first-principles consistency and accuracy rather than computation efficiency. The new technique must remain effective in handling typical high performance building scenarios and be progressive, where the accuracy of the solution should scale with the computational time.

4.3.1. Modifications to (Jensen) Photon Mapping

Simplification of Radiance-Components Expression, Global Photon Map

For consistency, the new rendering technique sacrifices the original method of using an approximate irradiance at each sampled point for calculation efficiency and reverts to the same four component radiance (Eq. 4-3) expression. At each sampled point, the radiance due to indirect illumination is evaluated using the global photon map directly, caustics are evaluated using the caustics photon map, and direct illumination by casting shadow rays towards light sources. The expression is thus organized in terms of three types of irradiance (direct, indirect, caustic) rather than reflection (diffuse or specular), since all three types of irradiance can be both diffuse or specular reflected at the sampled point.

$$\begin{aligned} L_r &= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_i(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \\ &= \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_{i,l}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \\ &+ \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_{i,d}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \\ &+ \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_{i,c}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \end{aligned}$$

Eq. 4-4

The BRDF is no longer split into diffuse or specular terms, since the implemented ideal specular (mirror), transparent, and Ward model (for opaque surfaces) allow the BRDF to be evaluated efficiently. At each sampled point x with surface normal $\vec{n}$, each photon contains the incident direction $\vec{\omega}'$. All parameters for evaluation the BRDFs are thus complete. In the case of specular and transparent materials, the BRDF itself is straightforward and already normalized (0-1). In the case of opaque surfaces, only the attenuation factor (and not the normalization factor) in the BRDF ($where\ \delta\ is\ the\ angle\ between\ surface\ normal\ \vec{n}\ and\ halfangle\ \vec{h} = (\vec{\omega}+\vec{\omega}')/\|\vec{\omega}+\vec{\omega}'\|$

Eq. 2-20) is used, since differential, not integral, values are being calculated.

In the original Photon Mapping technique, some paths such as reflected caustics are identified to be non-critical, and can be omitted to increase computation efficiency without sacrificing photo-realism. Since the focus is now on consistency and physical accuracy, all paths are explicitly considered in the new rendering technique to avoid underestimations. Figure 4-12 illustrates the difference in indirect illumination estimates between the original technique (left) and the new technique (right) where sample point irradiances are no longer approximated and all paths are considered.

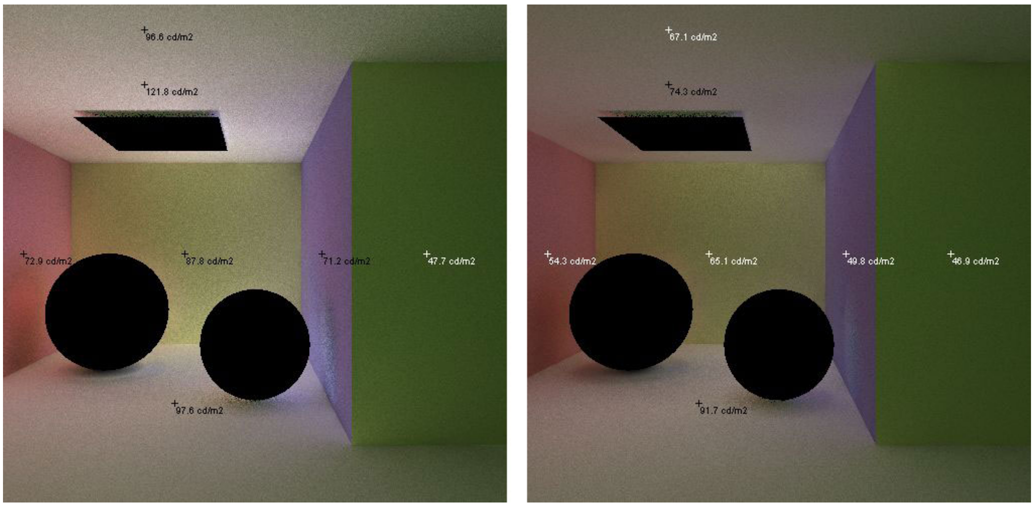


Figure 4-12 Indirect illumination in modified Photon Mapping (left), (Jensen) Photon Mapping (right)

While both images are of high photo-realism, the difference in absolute radiance values frequently exceeds the 8% perceivable-difference thresholds and illustrates the potential difference between the two approaches. It is important to note at this point that while the new technique removes photo-realism-motivated approximations, there is no guarantee on the physical accuracy of the new radiance estimates; additional empirical validation beyond the scope of this research has to be conducted, and is discussed in the future work section later (7.3.1).

	Point 1	Point 2	Point 3	Point 4	Point 5	Point 6	Point 7
(Jensen) Photon Mapping	67.1	74.3	54.3	65.1	49.8	46.9	91.7
Modified Photon Mapping	96.9	121.8	72.9	87.8	71.2	47.7	97.6
Percentage Difference	31%	39%	34%	26%	30%	2%	6%

Table 4-1 Sample radiance values from Figure 4-12

Photon Maps by Surface, Constant Radius instead of Constant Number of Photons

In the original technique, the radiance estimate uses a fixed number of photons and a varying area. This produces visually smoother results across adjacent points, especially if the density of the photons is low. Assuming the surface is locally flat around *x*, the radiance estimate in direction $\vec{\omega}$ is calculated by expanding a sphere around *x* until it contains *n* photons. The irradiance due to each photon *p* is then its flux ϕ_p divided by the projected area of the sphere, which is a disc. The reflected radiance is then the product of the irradiance and BRDF f_r:

$$L(x,\vec{\omega}) \approx \sum_{p=1}^{n} \frac{f_r(x,\vec{\omega},\vec{\omega}_p)\phi_p(x_p,\vec{\omega}_p)}{\pi r^2}$$

Eq. 4-5

The use of a sphere might result in including photons on other surfaces in the estimate, especially around corners of surfaces. To avoid this, the sphere can be compressed into a disc (Figure 4-13) as suggested by Jensen (Jensen, 2001).

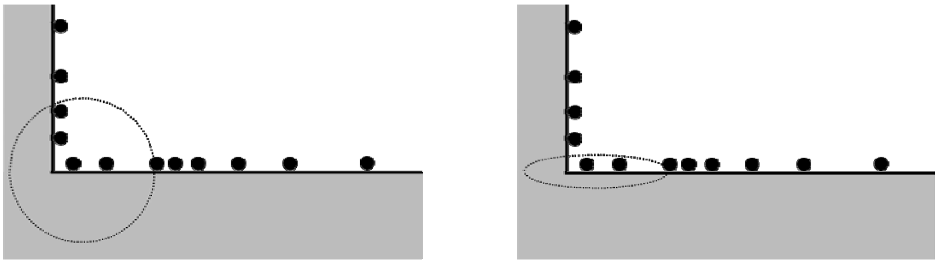


Figure 4-13. Compression from sphere (left) to disc (right) to avoid including wrong photons in the radiance estimate (Jensen, 2001)

Another simple solution, as implemented in the new tool, is to form photon maps by surface. In this case, only correct photons will be gathered, and the distance of each photon to the intersection point is simply the magnitude of the difference between the photon and intersection position vectors. Since the physically accurate technique is more concerned with consistency rather than visual effects, a constant radius rather than constant number of photons is used. In the latter, there is less visible variance when the photon density is sparse (smoother results), but the radiance might be over-estimated, such as when the sampling sphere is expanded from a low density point to a high density point in order to gather the required number of photons.

Given that lighting analysis in building design is usually interested in radiance or irradiance values on a fixed grid, a constant gathering radius is implemented in the new tool so that the radiance estimate at each point is consistent. The radius is a user-editable parameter, but heuristically set at 0.15m initially, representative of the 1-foot grid typically used in building analysis and design. The consequence is higher variance when the photon density is low, but this is acceptable since the premise, as mentioned earlier, is concerned with a consistent first principle-based approach instead of photo-realism. Additionally, the variance is lower when there are a high number of photons, which is typically the case. Typical scenes in this research use 0.3 to 4 million photons.

Area Corrections

In the original technique, the area estimate in Eq. 4-5 tends to be over-estimated at the corners of surfaces. However, since radiance estimates (using this equation) are not observed directly, the overall impact on the photo-realism of the generated image is small. Techniques such as filtering or calculating convex hulls are used to improve the area estimates; however, they might still have significant deviation from correct results.

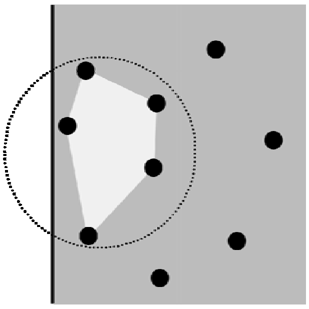


Figure 4-14. Using convex hull of photons to improve estimate of ΔA (Jensen, 2001)

Geometric solutions to general polygon surface – sphere intersections are difficult and computationally expensive. Fortunately, the degree of accuracy required by the building design context allows the use of a finite-element approach. Each surface is recursively subdivided into triangles and 256 barycentric-based random points are defined in each sub-triangle such that each point represents no more than 1% of the sampling area (projected disc area πr^2). When the sampling sphere is detected to intersect a surface edge, each barycentric point is tested if it is within the sphere. By counting the number of valid points and the sum of their areas, the area estimate will be consistently within ±5% accuracy regardless of the photon density. The technique is general and valid for all surface types, whereas previous approximations such as the convex-hull technique are not valid when the surface-sphere intersection area is non-convex.

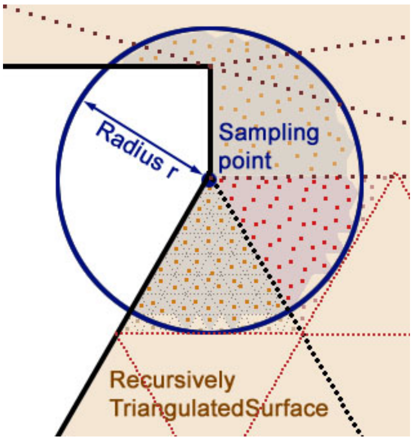


Figure 4-15 Finite element approach to estimating sampling area

Although a large surface may contain a large number of sub-triangles and consequently large numbers of points to be tested, the recursive nature of subdivision avoids the need to process all points. By adapting efficient sphere-edge intersection tests (Haines, 1989), it is easy to exclude testing ineligible sub-triangles and their recursive child-triangles. Optimized inside/outside testing introduced by the same author is also implemented to test all 256 random points in each eligible sub-triangle efficiently. The implementation in the new tool, before code optimization, processes in excess of 62,000 area estimates each second; which is a reasonable trade-off for the context of use. The radiance estimate $L(x,\vec{\omega})$ is then:

$$L(x,\vec{\omega}) \approx \frac{1}{A(x)}\sum_{p=1}^{n} f_r\left(x,\vec{\omega},\vec{\omega}_{\mathrm{p}}\right)\phi_{\mathrm{p}}\left(x_{\mathrm{p}},\vec{\omega}_{\mathrm{p}}\right)$$

Eq. 4-6

where $A(x)$ is the area of sampling sphere-surface intersection, which is estimated by the finite-element technique described above. Each of the 256 random points x_l in each eligible sub-triangle m with area A_m is tested $V_m \rightarrow \{0,1\}$ if it is within the specified radius from x, and inside the sub-triangle m.

$$A(x) \cong \sum_{m=1}^{n} \frac{A_m}{256}\sum_{l=1}^{256} V_m(x_l, x)$$

Eq. 4-7

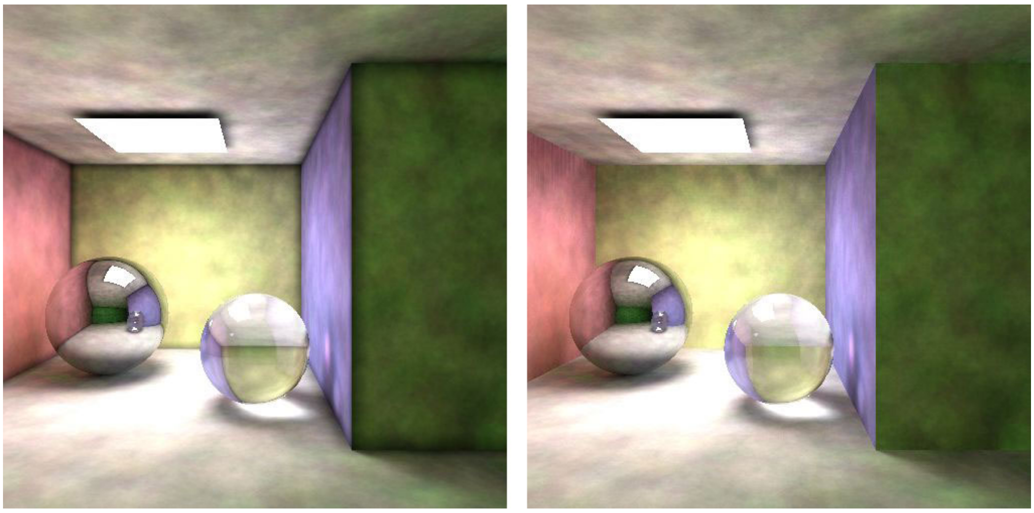


Figure 4-16 Direct visualization of $L(x,\vec{\omega})$ (Eq. 4-6) to show effect of approximated area (left), corrected area (right)

To visualize the effect this technique, the Photon Map is sampled directly in Figure 4-16 to show the radiance estimate$L(x,\vec{\omega})$. The area estimates near surface edges are now more accurate as compared to overestimation when using πr^2. The overestimation of area results in underestimating the irradiance and subsequently radiance. Consequently, the surface appears darker near surface edges. When the areas are corrected, the radiance is now consistently accurate regardless of surface location or shape. This implies that the irradiance estimates used (Figure 4-8) when solving the rendering equation are now of consistent accuracy, and would similarly yield better results (Figure 4-12). An additional benefit of using constant sampling radius and area correction is the possibility of using the photon maps directly, as discussed later (The light panel and spheres appear in Figure 4-16 due to caching of second (rendering) pass results).

4.3.2. Progressive Accuracy

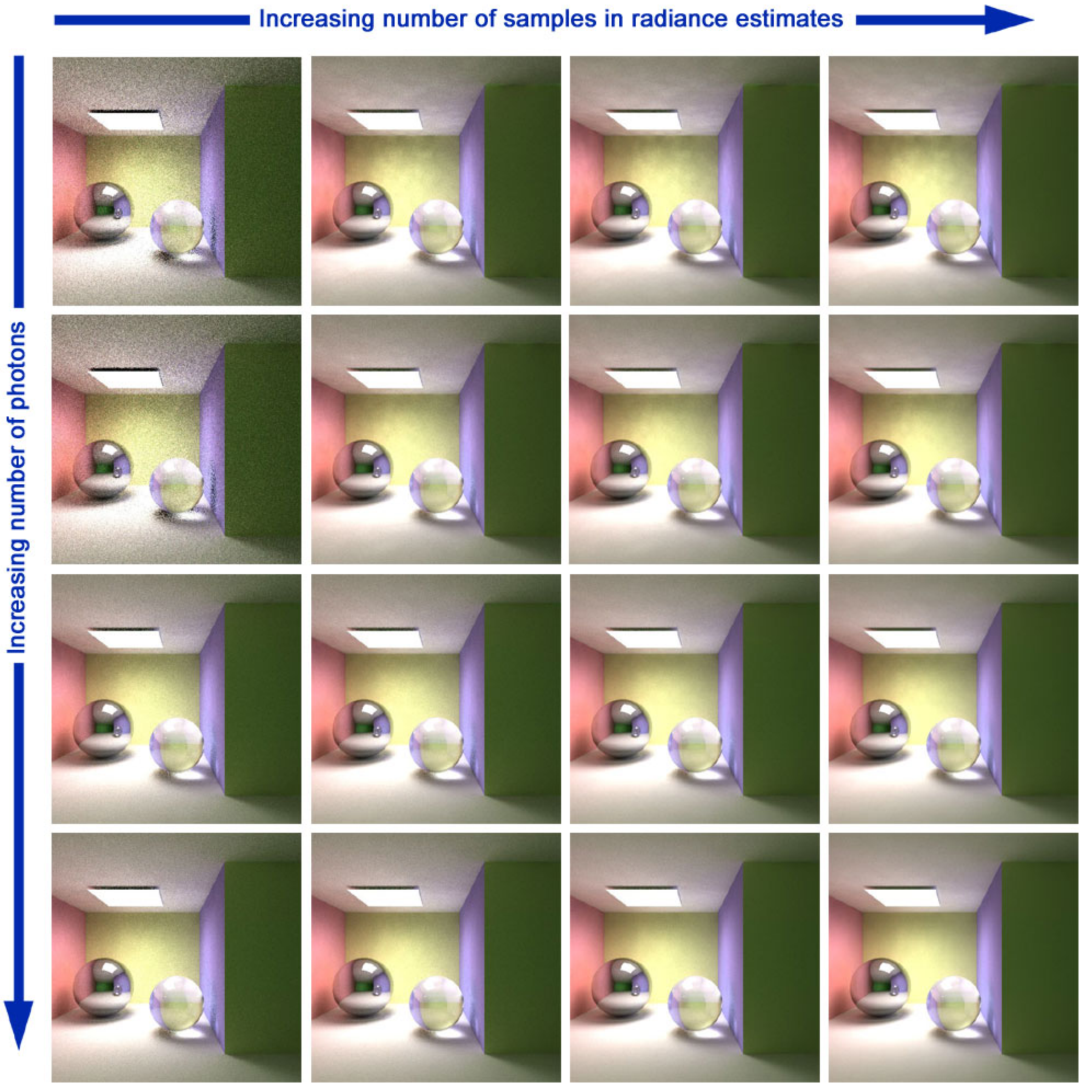

Increasing number of samples in radiance estimates
Increasing number of photons

Figure 4-17 Effects of increasing number of photons and number of samples in radiance estimates

The Photon Mapping technique lends itself naturally to progressive simulation where the results can be progressively refined with additional processing (Hachisuka, Ogaki, & Jensen, 2008). Since each photon represents a portion of the original radiant flux from light sources, additional photons can be traced independently and simply added to scaled previous results. This independent identically distributed characteristic where results can be progressively improved by more simply adding more samples applies to both the photon tracing and radiance estimating parts of the technique.

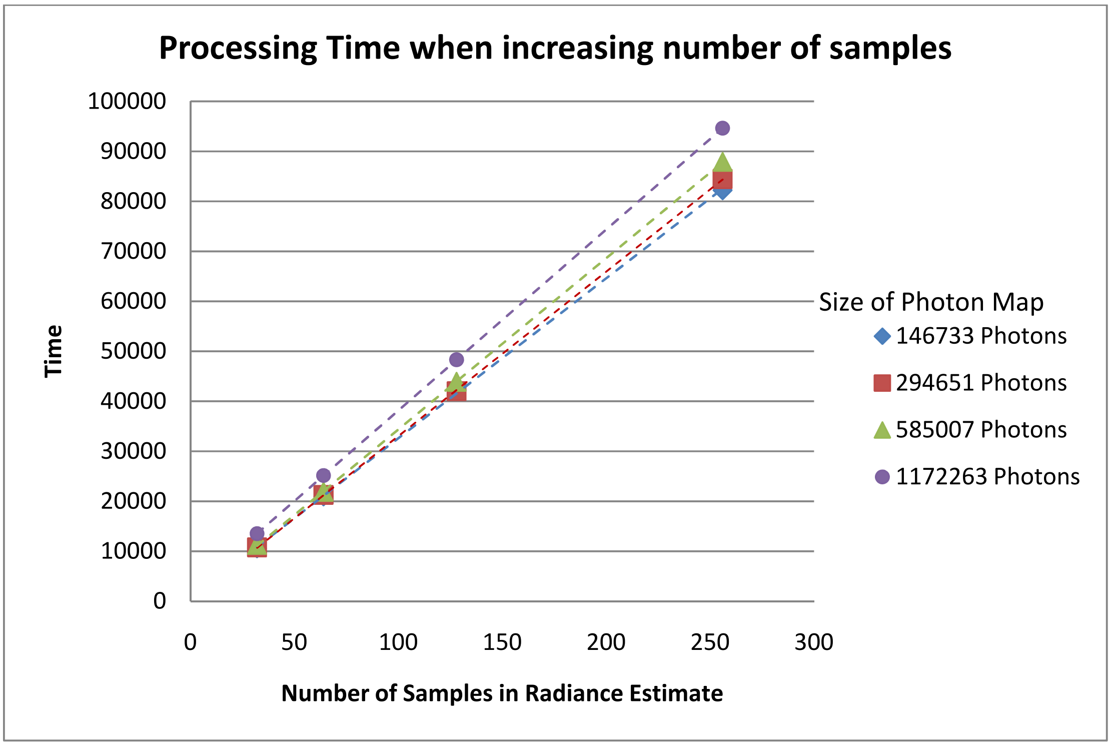


Figure 4-18 Processing times of images in Figure 4-17

Figure 4-17 shows the effect of increasing the number of photons and increasing the number of samples in radiance estimates. Both strategies yield visibly more photo-realistic as well as more accurate results. More photons implies higher photon densities and corresponding more accurate irradiance estimates at all points in the photon map, and the estimated radiance (Eq. 4-6) tends towards actual radiance as the number of samples tends towards infinity. Figure 4-18 makes the case for using more photons rather than more samples. While overall processing time increases (linearly) with the number of samples in the radiance estimate, the time does not vary much when there are more photons in the photon maps. The overall processing time consists of three parts: photon tracing, photon-tree balancing, and radiance estimates for individual pixels in the desired image.

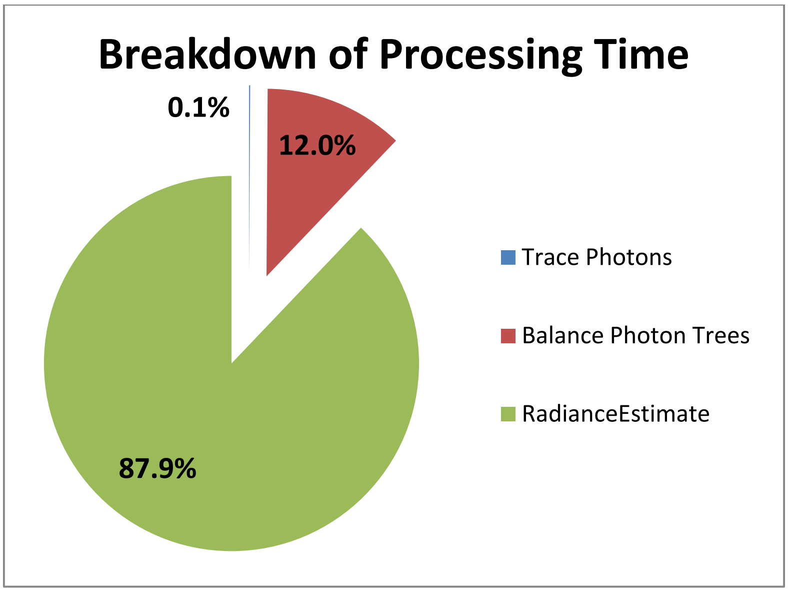


Figure 4-19 Breakdown of processing time

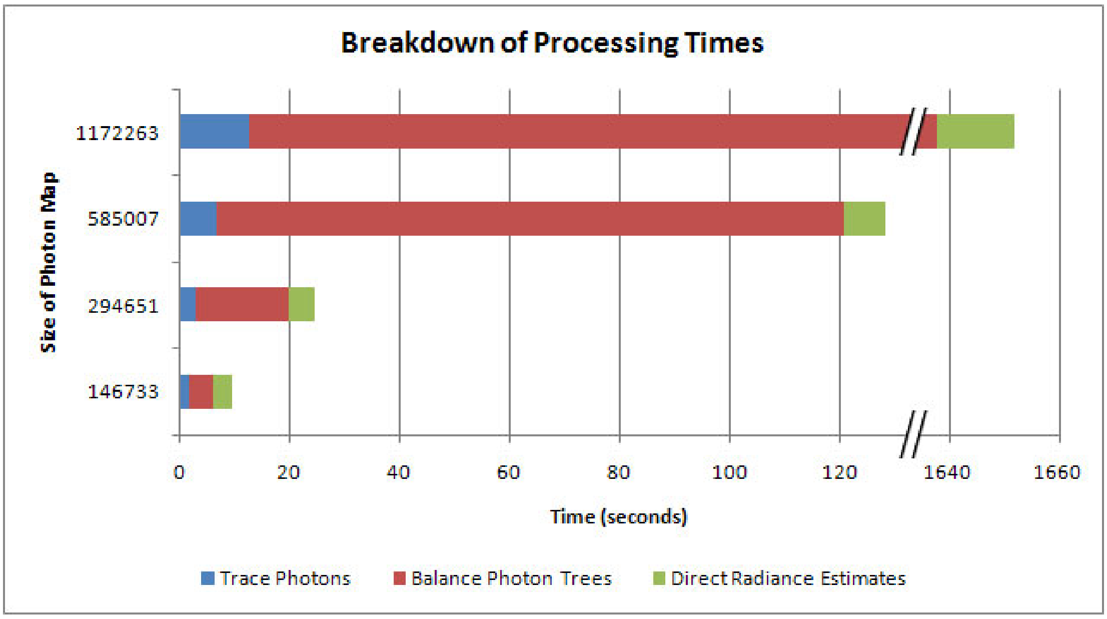
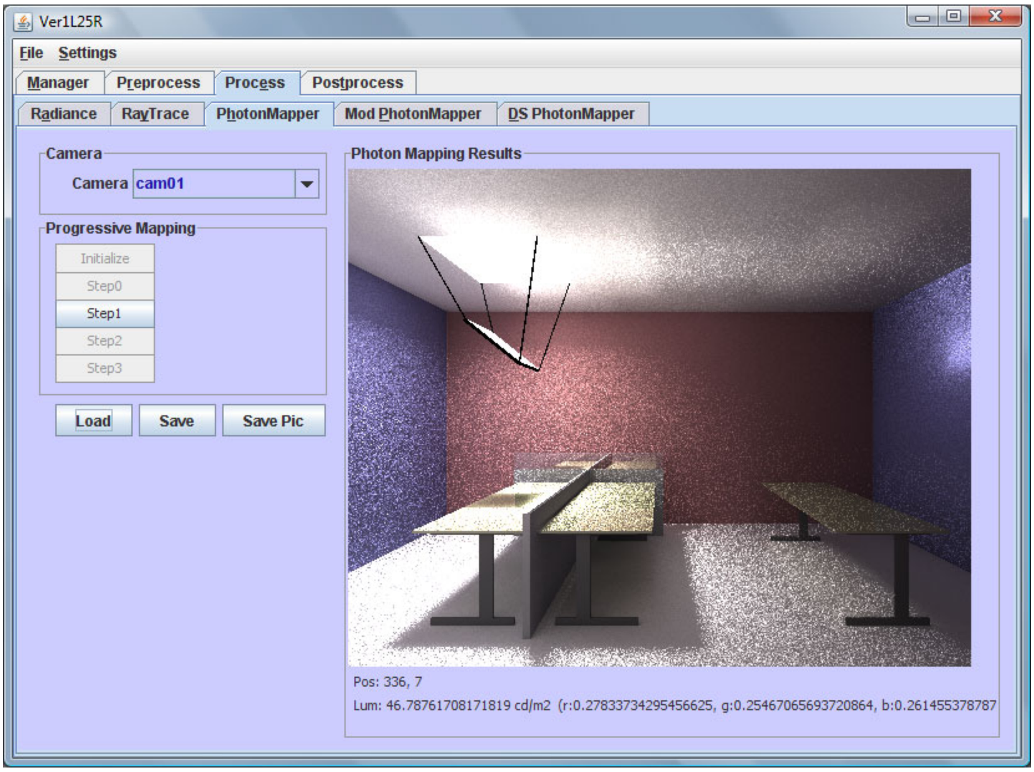


Figure 4-20 Comparison of processing times

Since a fixed sampling radius is used, there is overall less photons to process when more photons are considered at each sampling point, as compared to more sampling points, each with slightly less photons involved. Figure 4-19 (1 million photons, 32 samples in radiance estimate, bottom-left image in Figure 4-17) shows that most of the processing time is spent on radiance estimates.



```
Pos: 336, 7
Lum: 46.78761708171819 cd/m2  (r:0.27833734295456625, g:0.25467065693720864, b:0.261455378787
```

Figure 4-21 Photon Mapping UI in new tool

Progressive photon mapping is implemented in the new tool primarily by increasing the number of photons, although the number of samples is also nominally increased at each step. This feature is implemented as a simple series of buttons in the UI (Figure 4-21), where the user is presented with “Step” buttons progressively to increase the accuracy of the simulation by “steps”.

The (backwards) ray tracing pass, termed “initialization” in the UI, has to be conducted first. In this pass, direct diffuse and specular (recursive) reflected surface intersections for each pixel, surface sub-division material BRDFs and probability functions, as well as lamp distribution functions, are pre-computed and cached. Ray tracing is then used to evaluate the direct illumination. Subsequently, in each step where the user desired increased accuracy, additional photons are emitted and traced to yield the next set of simulation results.

4.3.3. Importance Sampling to Handle Typical High Performance Building Designs

Importance sampling is a technique to direct ray tracing effort towards areas of importance to the overall result (Jensen, 1995; Christensen, 1999). This concept is implemented in the new rendering engine to deal with highly-reflected irradiance conditions typical in high performance building designs. While Photon Mapping generally performs well (Figure 4-11) in all conditions, a large number of photons is required to handle highly diffuse scenes well. In such cases, highly reflective surfaces in the form of light re-directors typically account for only a small portion of the total surface area photons interact with, but have significant impact on the overall irradiance and radiance in the scene. This means that even when large numbers of photons are used, only a small number will cover the desired paths.

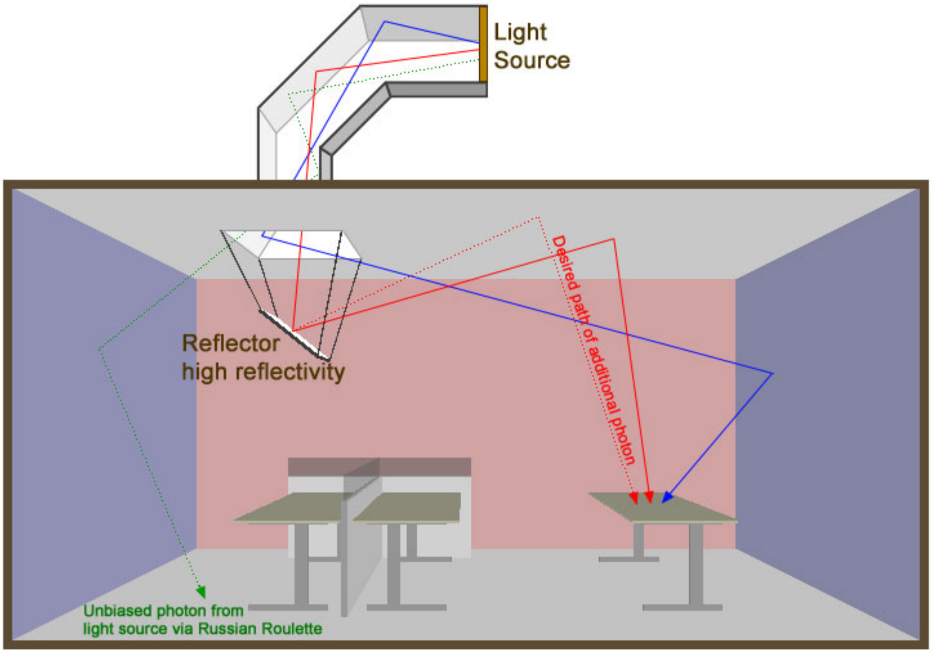

Light
Source
Reflector
high reflectivity
Desired path of additional photon
Unbiased photon from
light source via Russian Roulette

Figure 4-22 Power-based priority queue for emitting additional photons

Importance sampling in the form of a power-based priority queue is implemented to efficiently increase photon density in the desired regions, as well as even out the difference in photon powers. Using the example of the experimental case (Figure 4-22), only a small number of photons will fall upon, and be reflected by, the highly reflective light re-director. To improve the accuracy of the simulation results, it is desirable for additional photon to be emitted (as per progressive technique described earlier) and have paths including reflection by the reflector (red dashed line in Figure 4-22). However, in the original Russian Roulette technique, there is no guarantee that the paths of the additional photons will include reflection by the reflector.

The preferred paths can be identified by the magnitude of the exitant radiance, which in turn is the product of incident irradiance and overall surface reflectance (integral of BRDF). While evaluating irradiance at all photon-surface intersections is computationally expensive and cannot be performed until all photons have been traced, irradiance is related to the radiant flux of each incident photon at the location. Exitant radiance, and consequently the degree of preference of paths $P(x)$ including points near the location x, can thus be heuristically correlated with the product of radiant flux (power) $\Phi_{\mathrm{p,x}}$ of the photon p arriving at x and surface reflectance ρ.

$$P(x) = \rho \cdot \Phi_{\mathrm{p,x}}$$

Eq. 4-8

A priority queue, in the form of a heap, is maintained to track this degree of preference for all photons during the usual photon tracing stage. As a second pass, additional photons are emitted from the most important points as identified by the photons with the highest degrees of preference, before being scattered by the same Russian Roulette technique. The power of photons originally reflected from these points have to be scaled accordingly so that the Monte Carlo conditions still hold, and the photon map maintains overall correctness. The additional photons introduced by this technique are not newly emitted photons (from light sources); they are conceptually part of the same group of photons reflected from a single parent photon (the photon identified by the priority queue). In other words, instead of having a single Russian Roulette reflected photon from a high-preference photon, there are now two reflected photons, each carrying half the original power, as attenuated by the BRDF. By concentrating on surfaces that have more impact on the overall lighting, this technique yields accurate results more efficiently.

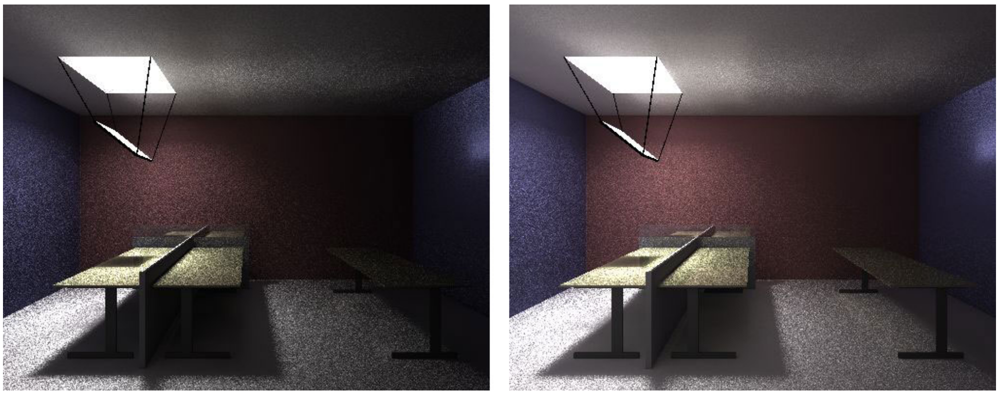


Figure 4-23 Conventional Russian Roulette (left), power-prioritized technique (right)

In the figure above, both sets have the same number of photons emitted from the light source. In the right image, the power-prioritized technique identifies the photons incident on the high reflectance reflector and introduces photons from these points. Consequently the highlight caused by the reflector on the ceiling is visibly better defined, and the right side of the room appears brighter (since radiance estimates at these points now include samples on a brighter ceiling).

One modification and two additional data structures are introduced to implement this power-prioritized technique. To avoid having to rebalance the photon map kd-tree when adding photons in this technique, the conditions of a median split is relaxed. The allowable size on each side of the split is increased slightly, so that a photon can be inserted quickly (via a quick sort algorithm). When even this increased size is exceeded, the entire side is median-split into axis-orientated left and right children according to the usual kd-tree algorithm. While this decreases the theoretical efficiency of the kd-tree, the practical impact is minimal since the number of photons to be quick-inserted is small. Alternatively, if the power-prioritized pass is not offered as a separate feature but integrated with the normal Russian Roulette pass, balancing of the kd-tree can be performed only upon completion of both passes. In this case, there is no need to implement the quick-insert algorithm.

The additional data structures are the priority queue and a photon-relations map. The priority queue is a simple heap where each node contains the photon index and the degree of preference, expressed as a float. The photon-relations map stores all photon paths, with references to the photon objects for fast access when modifying photon powers. In the power-prioritized pass, the indexes of preferred photons (parent photons) are retrieved from the heap. The relevant node in the photon-relations map is retrieved and the power of all children branches halved. The additional photon (with half the parent photon radiant flux before BRDF attenuation) and subsequent children (as determined by Russian Roulette), is then appended as a branch beneath the parent photon node.

4.4. Fast Visualizations

A beneficial side effect of the techniques implemented to progressively increase the accuracy, and to deal with the typical high performance design scenarios, is that the photon maps tend to contain large numbers of photons, and have better (more even by having more photons each with fractional powers, except in cases of caustics where concentration of the photons is desirable) photon densities. Since the radiance estimate (Eq. 4-6) tends towards correctness with large number of photons, this implies that the photon map can be used directly, with acceptable accuracy, once the photon density crosses a threshold.

The benefit of using the photon map directly to estimate radiance for each pixel point is evident in Figure 4-19, which shows that the majority of the processing time goes towards evaluating radiance estimates. Using the photon map directly reduces the processing time by at least (usually much more since other overheads are also reduced) n times, where *n* is the number of previously specified number of sub-samples. The rendering times (excluding photon tracing and balancing) for the left column images in Figure 4-17 are summarized in the table below.

	Image 1 **146664 photons**	**Image 2** **290683 photons**	**Image 3** **587261 photons**	**Image 4** **1169356 photons**
32 Sub-Samples per Radiance Estimate	10579s	10749s	11154s	11868s
Direct sampling	3.5s	4.7s	7.5s	14.1s

Table 4-2 Comparison of radiance estimate processing times

The same set of images, using direct sampling instead of sub-sampling, is shown below, and two indicative point radiance values are compared against those from a high photon number, high sub-sample number reference image in Table 4-3.

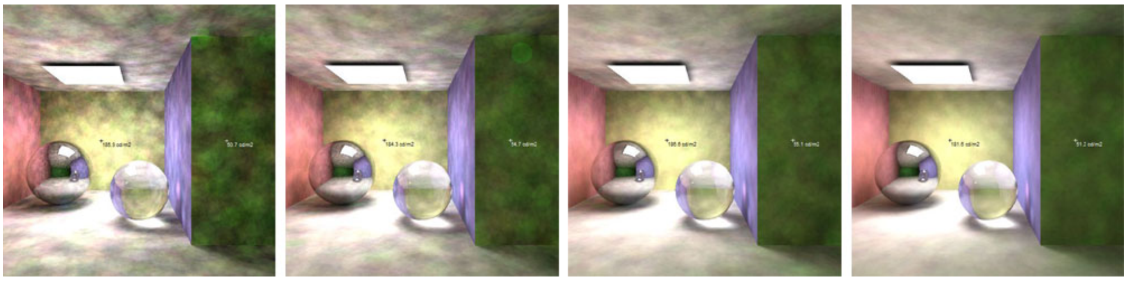


Figure 4-24 Direct sampling of photon maps with increasing numbers of photons (left to right)

	Reference Case	Image 1 146664 photons	Image 2 290683 photons	Image 3 587261 photons	Image 4 1169356 photons
Point 1 (cd/m^2)	178.9	185.9	184.3	186.6	181.6
Point 2 (cd/m^2)	46.5	50.7	54.7	55.1	51.2
Percentage Difference	-	5%	6%	7%	3%

Table 4-3 Comparison of indicative radiance values with reference case

The poor visual quality of the first three images in Figure 4-24 suggests that the photon density is below the threshold where photon maps could be used directly. Consequently, there is no immediate observable trend in the percentage difference from the high quality reference case. However, the radiance values show relatively good coherence with the high quality reference case, without exceeding 7% difference in value. In the case of Image 4 which contains excess of a million photons, the difference is a mere 3%. This suggests that direct sampling of the photon map could be used when there is sufficiently high photon density, or when some loss in accuracy is acceptable for significant decrease in computation time. While the threshold for photon density is heuristically determined at this point by examining the visual quality of the direct sample images, it is possible to implement a limited region of (conventional) sub-sampled radiance estimates to gauge the quality of the direct sampled results. Sub-sampled radiance estimates should be based on the average values (or Gaussian filtered) over a small region, instead of a single pixel, to avoid outliers caused by variance.

Within adaptive-iterative design processes, metrics are used comparatively to evaluate the impact of varying design strategies, and the results should ideally be available within short time spans as not to disrupt design workflows. Additionally, the operative information required is typically grid values (checking working-plane illuminance, luminance ratios, and lighting distribution); photo-realistic imagery is less important. Fixed-radius direct sampling to achieve fast visualizations is suitable for such scenarios since the degree of inaccuracy would be similar between two similar models using similar number of photons, allowing the values between the two sets of results to be comparatively used. In the third scenario where differences between models might be significant, care should be exercised to ensure that the photon densities are high enough.

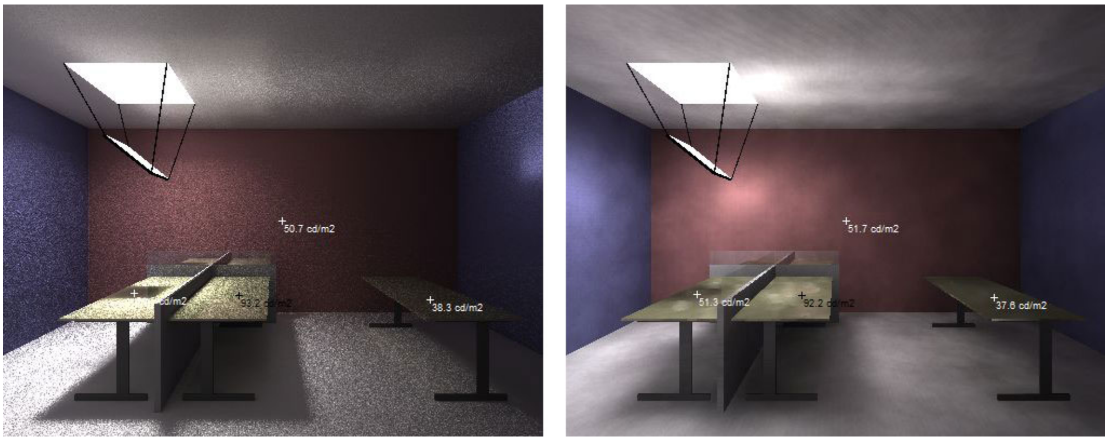


Figure 4-25 Comparison of radiance values, sampled radiance estimates (left), direct sampling (right)

The issue of photon densities is largely averted when the power-based importance sampling technique is implemented, since the technique increases overall photon density. In Figure 4-25, the average difference between the sub-sampled radiance estimates and direct sample values for indicative points on the work surfaces (tabletops) and the wall is only 1.3%, while each point is no more than 2% different, well within the perceivable-difference threshold of 8%.

	Point 1 (Left Desk)	**Point 1 (Mid Desk)**	**Point 1 (Right Desk)**	**Point 1 (Wall)**	
Sub-Sampled Radiance Estimate (cd/m^2)	51.5	93.2	38.3	50.7	
Direct sampling (cd/m^2)	51.3	92.2	37.6	51.7	**Average % Difference**
Percentage Difference (Absolute difference)	**0.4%**	**1.1%**	**1.8%**	**2.0%**	**1.3%**

Table 4-4 Comparison of indicative radiance values when power-based importance sampling implemented

5. Computational Formulation of Lighting Benchmarks

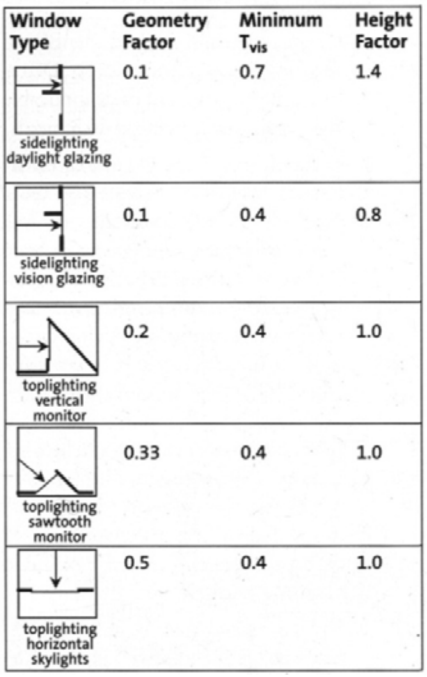
Typical benchmarks pertinent to typical high performance building designs are formulated as computable for subsequent implementation in new tool.

5.1. LEED EQ. 8.1

LEED Credit EQ 8.1 is a benchmark that quantifies the amount of daylight availability in regularly occupied areas of a building. The benchmark allows three methods of calculation: by glazing factors, computer simulation and actual on-site measurements. The last case is obviously for built projects and not considered here. Glazing factors (Eq. 5-1) is an estimation of day lighting conditions based on window positions and visible transmittance for overcast sky conditions.

$$Glazing\ Factor = \frac{Window\ Area}{Floor\ area} \times Window\ GF \times \frac{Actual\ T_{vis}}{Min\ \ T_{vis}} \times Window\ HF$$

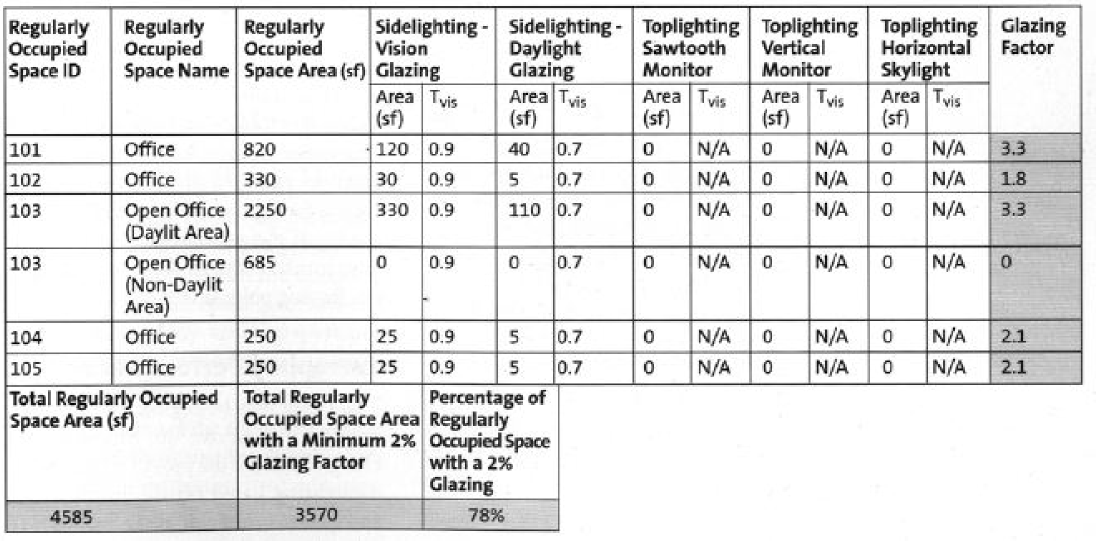
Eq. 5-1



Window Type	Geometry Factor	Minimum T_{vis}	Height Factor
sidelighting daylight glazing	0.1	0.7	1.4
sidelighting vision glazing	0.1	0.4	0.8
toplighting vertical monitor	0.2	0.4	1.0
toplighting sawtooth monitor	0.33	0.4	1.0
toplighting horizontal skylights	0.5	0.4	1.0

Table 5-1 Geometry Factor (GF) and Height Factor (HF) Definitions (U.S. Green Building Council, 2005)

Glazing factors do not account for varying sky conditions at different locations, room geometry, or varying reflectance values across different surfaces in the room. Glazing factors as a performance indicator thus perform poorly when the room is non-convex, has high aspect ratios, or when the surfaces encompass a wide range of reflectance values. The advantage of glazing factors is that it can be computed easily.



Regularly Occupied Space ID	Regularly Occupied Space Name	Regularly Occupied Space Area (sf)	Sidelighting - Vision Glazing		Sidelighting - Daylight Glazing		Toplighting Sawtooth Monitor		Toplighting Vertical Monitor		Toplighting Horizontal Skylight		Glazing Factor
			Area (sf)	T_{vis}	Area (sf)	T_{vis}	Area (sf)	T_{vis}	Area (sf)	T_{vis}	Area (sf)	T_{vis}	
101	Office	820	120	0.9	40	0.7	0	N/A	0	N/A	0	N/A	3.3
102	Office	330	30	0.9	5	0.7	0	N/A	0	N/A	0	N/A	1.8
103	Open Office (Daylit Area)	2250	330	0.9	110	0.7	0	N/A	0	N/A	0	N/A	3.3
103	Open Office (Non-Daylit Area)	685	0	0.9	0	0.7	0	N/A	0	N/A	0	N/A	0
104	Office	250	25	0.9	5	0.7	0	N/A	0	N/A	0	N/A	2.1
105	Office	250	25	0.9	5	0.7	0	N/A	0	N/A	0	N/A	2.1

Total Regularly Occupied Space Area (sf)	Total Regularly Occupied Space Area with a Minimum 2% Glazing Factor	Percentage of Regularly Occupied Space with a 2% Glazing
4585	3570	78%

Table 5-2 Example of LEED Credit EQ 8.1 tabulation (U.S. Green Building Council, 2005)

The procedure listed in the LEED documentation describes the inspection of design drawings to identify external windows and skylights, determine the classification of windows, and calculate their areas. Vertical windows extending beyond the 2’6” to 7’6” (762mm to 2286 mm) vertical height range have to be further divided into non-contributing (portions below 2’6”), vision glazing (portions between 2’6” to 7’6”), and daylighting glazing (portions above 7’6”). Once the areas are obtained, together with the material properties, the areas are simply tabulated via a spreadsheet that performs the Glazing Factor equation.

5.1.1. Implementation

The calculation of the glazing factor requires the determination of the type for each window according to Table 5-1, the relevant areas as well as the tabulation of visible transmittance values. In lighting tools that contain building geometry information and material data, this calculation can be done with relative ease. The steps in an algorithm that does so are:

Step 1: find the list of occupied spaces

Step 2: find the list of windows in each space

Step 3: determine the window types (subdivide the window if necessary)

Step 4: retrieve T_{vis} and calculate the GF of the windows

Step 5: tabulate the GFs in space and determine if equal or greater than 2%

Step 6: tabulate the eligible floor area in the building and determine if equal or greater than 75%

Most of the steps in the algorithm involve only logistical tasks, such as the sorting and tabulation of occupied spaces and windows. Only Step 3 involves actual computation, but this can be done with relative ease. By retrieving the geometry of the window and the space it is in, the height, orientation, and consequently type, of the window can be calculated very quickly. When implemented (5.3 below), the algorithm calculates the LEED credit instantaneously.

5.2. LEED EQ. 8.2

LEED Credit EQ 8.2 is a benchmark that quantifies the percentage of occupied spaces that have exterior view. Specifically, the credit is awarded when 90% of all regularly occupied areas have a view to an exterior vision window. Vision windows are defined as portions of exterior windows between 2’6” (762mm) and 7’6” (2286mm) above finish floor levels of each room considered. The credit can be formulated as thus:

$$EQ_{8.2} = \begin{cases} 1 \mid \sum (V(Room_n) \div Area_n) \geq (0.9 \times Area_{total}) \\ 0 \mid otherwise \end{cases}$$

Eq. 5-2

$where\, Room_n \in Regulary\ Occupied\ Spaces\ in\ Building$
$Area_n = Floor\ Area\ of\ Room_n$
$Area_{total} = \sum Area_n$
$V(Room_n) = Floor\ Area\ of\ Room_n\ with\ view\ to\ Vision\ Window$

The documentation also stipulates that the entire areas of single-occupied rooms are eligible for consideration if 75% or more of the room area has a view to some vision window; otherwise the actual area with views to vision windows is to be used. For multi-occupant rooms, actual areas with view to vision windows are used. This can be formulated as:

$$
\begin{aligned}
V(Room_n) &= \begin{cases} Area_n & |\ single\ occupant\ \wedge\ \left(V_{actual}(Room_n) \geq (0.75 \times Room_n)\right) \\ V_{actual}(Room_n) & |\ single\ occupant\ \wedge\ \left(V_{actual}(Room_n) < (0.75 \times Room_n)\right) \\ V_{actual}(Room_n) & |\ multi\ occupant \end{cases} \\
&= \begin{cases} Area_n & |\ single\ occupant\ \wedge\ \left(V_{actual}(Room_n) \geq (0.75 \times Room_n)\right) \\ V_{actual}(Room_n) & |\ otherwise \end{cases}
\end{aligned}
$$

Eq. 5-3

where $V_{actual}(Room_n)$ = Actual Calculated Floor Area of $Room_n$ with view to Vision Window

The procedure for determining the floor area with view to vision window in each room is described graphically (Figure 5-1). The documentation describes a 2-step process; sightlines are drawn on plan to calculate an interim area with view to perimeter windows, sightlines are then drawn for each eligible window in representative section to confirm if unobstructed view conforms to vision window definitions. The line of sight heights used in sections are defined to be 42" (1067 mm) above finish floor levels representing average seated eye heights.

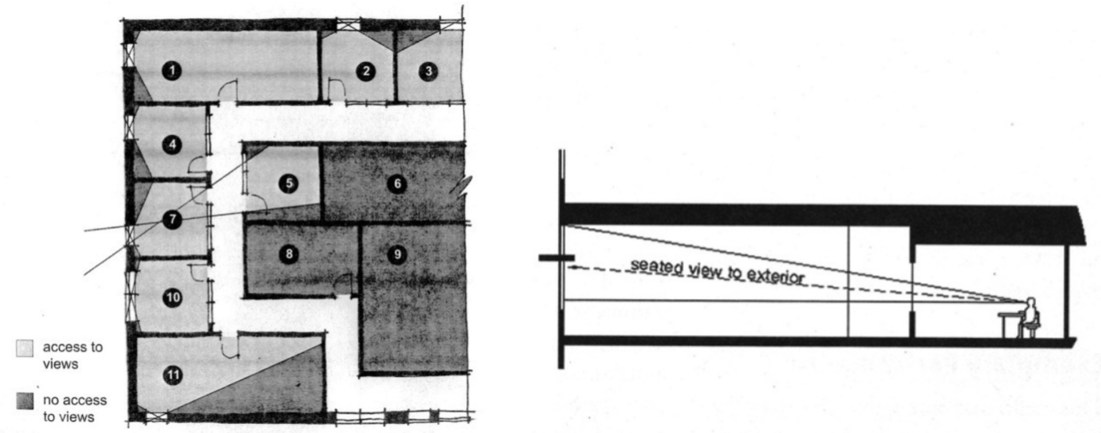
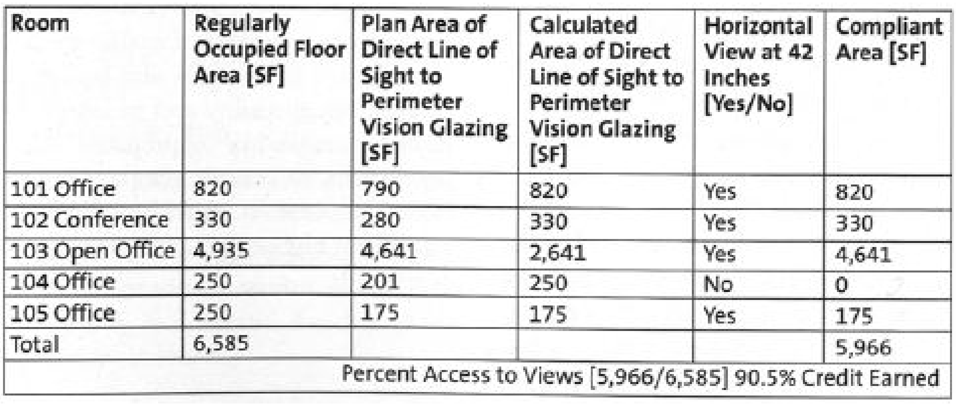


Figure 5-1 Drawing sightlines on plan (left) and section (right) to determine view to vision window. (USGBC, 2005)



Room	Regularly Occupied Floor Area [SF]	Plan Area of Direct Line of Sight to Perimeter Vision Glazing [SF]	Calculated Area of Direct Line of Sight to Perimeter Vision Glazing [SF]	Horizontal View at 42 Inches [Yes/No]	Compliant Area [SF]
101 Office	820	790	820	Yes	820
102 Conference	330	280	330	Yes	330
103 Open Office	4,935	4,641	2,641	Yes	4,641
104 Office	250	201	250	No	0
105 Office	250	175	175	Yes	175
Total	6,585				5,966
Percent Access to Views [5,966/6,585] 90.5% Credit Earned					

Table 5-3 Example of LEED Credit EQ 8.2 tabulation (U.S. Green Building Council, 2005)

While the narrative and graphical description for calculating the availability of vision windows suggests a geometric approach, the formulation of the procedures shows the possibility of a finite-element approach. While further analysis is required to determine which is computationally faster, the latter lends to faster computer implementation. In this case, the floor plane is discretized and each point checked for access to vision windows. The procedure can be formulated as:

$$V_{actual}(Room_n) = \sum_{i=0}^{p} A(i)$$

Eq. 5-4

$where\ i\ \in \{\, p\ points\ in\ Room_n\ 42"\ above\ finish\ floor\ level\}$
$A(i) = Availability\ of\ view\ to\ vision\ window\ at\ point\ i$

By enumerating the vision windows in the building as $W = \{vision\ windows,\ w_{1,2,\ldots l}\}$, $A(i)$ can be defined more accurately as $A(i, W)$, which expresses the availability of view to any of the windows in set W. $A(i, W)$ can then be expressed as a recursive algorithm:

$$A(i, W) = \big(E(i, w_l) \vee A(i, \{W - w_l\})\big)$$

Eq. 5-5

$where\ E(i, w_l) = Availability\ of\ view\ to\ w_l\ from\ point\ i$

$E(i, w_l)$ can be evaluated by standard ray-tracing by testing for opaque obstructions between point i and points on vision window w_l.

5.2.1. Improvements

Following the preliminary definition of $E(i, w_l)$, two shortcomings are obvious in the current procedure stipulated by LEED. First, infinitesimally small areas of vision windows constitute a view to the exterior. Second, there is no consideration of the combined visual effects from several windows. To address these two issues and improve Credit EQ 8.2 as a benchmark of exterior view availability, some changes can be introduced with minimal change in computational expense.

The definition of vision windows reflects an attempt to quantify typical visual behavior and the areas of visual sensitivity. Using the same 1067mm eye height, the current vision window height limits describe a 22˚ field of vision (10˚ above horizon, 12˚ below horizon) 4m away from the window, which is roughly consistent with cone of temporal vision sensitivity. However, such visual behavior consideration becomes moot with the current stipulated calculation method. Regardless, the current definition also becomes problematic when the distances increase. At 10m (a reasonable distance in large buildings), the current limits describe only a 7˚ field of vision, which is obviously too narrow for exterior context awareness.

The use of visual angles is thus advantageous. By stipulating a minimally required viewing angle to constitute a valid view to the exterior, infinitesimally small areas become invalidated. Furthermore, the description of visual behavior becomes consistent regardless of distance; a larger vision window is required for deep spaces. On the other hand, window sections above or below the current limits or even finish floor levels become eligible if there is a direct view. This actually increases flexibility for design. Pending more detailed investigations, separate horizontal and vertical minimal viewing angles can be used to describe asymmetrical visual behavior.

Steradians subtended by each window can be used together with the recommended visual angles to better describe the visual impact of windows. A minimal limit for individual and cumulative steradians subtended by the windows can then be used to eliminate views that are too restrictive to constitute exterior awareness (such as narrow slits), as well as account for combined effects of multiple views (a series of slit-like views may actually allow for exterior awareness. In the latter case, an additional visual angle between candidate views may be necessary to define acceptable cases. Using the same example of multiple slit-like views, exterior awareness may be negated when the slits are too far apart and not present together a coherent view.

5.2.2. Implementation

Since Credit EQ 8.2 is granted upon achieving 90% of the building floor area, the calculations only have to achieve accuracy to the nearest percentage. Correspondingly, $V_{actual}(Room_n)$ (Eq. 5-4), only have to achieve this accuracy and the number of samples per room, p can be limited. Likewise, if visual angles are implemented to improve the accuracy of the benchmark, the number of samples in $E(i, w_l)$ (Eq. 5-5) to be taken for each window can be correspondingly limited.

To reduce the computational load in evaluating $E(i, w_l)$ by ray-tracing, the set of external windows is pre-processed into a smaller set of eligible vision windows as defined by the more accurate viewing angles. A smaller list of possible obstructions is obtained from the set of all building surfaces and organized as a surface area heuristic (SAH) cost-optimized kd-tree (Bentley, 1975; MacDonald & Booth, 1990). Since these two processes are essentially linear to the number of elements, the additional computational load is near negligible. The steps in the algorithm are:

Step 1: find the list of occupied spaces

Step 2: generate the list of points in each space

Step 3: generate the list of candidate rays from the points to vision windows

Step 4: trace the candidate rays for obstruction

Step 5: calculate the steradians subtended by external views

Step 6: tabulate the floor area with view to exterior

Step 7: determine if eligible floor area in building is equal or greater than 90%

Like the previous calculator, this algorithm calculates Credit EQ 8.2 within seconds as presented later.

5.3. Benchmarking Speed of Computations

An experiment is conducted to compare the speeds of manual and automatic calculation; this is to illustrate the time and effort reductions achievable by implementing the algorithms above. A series of building designs with varying levels of complexity are prepared (Figure 5-2), and the time taken to perform both LEED credit calculations measured. While it is obvious that automated computing would be faster, the experiment establishes the magnitude of time savings. The pairs of manual-automatic calculation also serve to validate the accuracy of the algorithms.

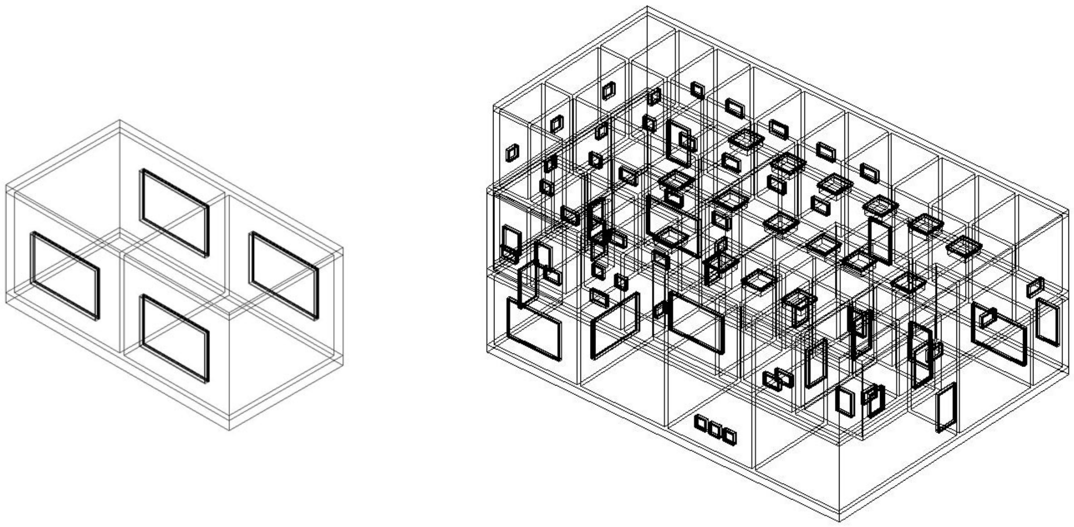


Figure 5-2 Designs with varying levels of complexity (not to scale)

Automated Calculation	Manual Calculation
• Well-formed BIM available	• Well-formed CAD drawings (plans and sections) All material properties available
• Typical User (no prior training)	• Expert User (familiar with LEED submittals) Calculation spreadsheets available

Table 5-4 Comparison of conditions between automatic and manual calculations

To limit the comparison to the effect of computation efforts rather than information gathering (such as locating drawings and researching material properties), both methods of calculation are given complete and well-formed information. Similarly, familiarity with the LEED credit calculation methods and referenced values is assumed for the manual case. In additional, the manual case also uses a prepared calculation spreadsheet; the user only has to fill in the necessary design data and the spreadsheet does the tabulation. Redundant efforts due to errors in manual calculation are not included. The experiment thus only assesses the difference between the procedures described within the LEED documentation, and the computation formulations.

	LEED Credit EQ 8.1		LEED Credit EQ 8.2	
Model Description	Manual	Automated	Manual	Automated
2 Zones 4 Windows	148s	0.00s	276s	0.12s
3 Zones 6 Windows 2 Skylights	469s	0.00s	570s	0.42s
4 Zones 8 Windows 2 Skylights	477s	0.00s	567s	0.57s
8 Zones 16 Windows 4 Skylights	701s	0.00s	1043s	0.58s
16 Zones 32 Windows 8 Skylights	966s	0.01s	3127s	7.29s
32 Zones 64 Windows 16 Skylights	2079s	0.01s	7536s	77.85s

Table 5-5 Tabulation of times taken to calculate LEED credits

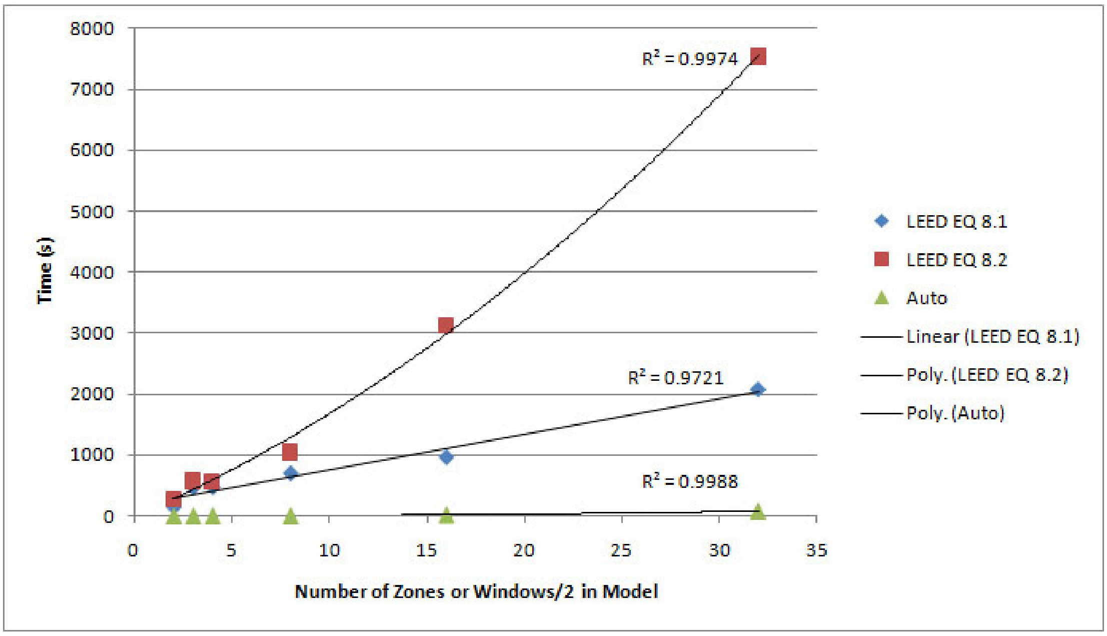


Figure 5-3 Trend of calculation times

While the set of calculation times is small, there is a perceivable trend relating the calculation times and the complexity of the model. Manual calculation of LEED Credit EQ 8.1 tends to be directly proportional to the number of zones and windows, since the tabulation requires obtaining the areas of all zones and windows; the increase in number of zones and windows does not complicate the calculation of each zone or window area. While in reality some complexity might arise from the logistical effort in identifying the zone or window in the drawings, this is not considered in the experiment; a set of well organized drawings was used for the manual calculation. In the case of LEED Credit EQ 8.2 manual calculation however, there is a noticeable time increase per zone or window to perform the calculation. This is because the view-out calculations of interior spaces can now include view-paths through multiple windows; the identification and determination of boundary view-paths become increasingly complex as the number of zone or windows increases. During the experiment, mistakes were often made and require significant effort to correct. While such correction times were eventually not included in the tabulation, the overall calculation time is still significant.

In contrast, automated calculation is significantly faster. In the case of LEED Credit EQ 8.1, the time taken is negligible, since the computation is mainly logistical processing as mentioned earlier. Even the LEED Credit 8.2 calculation times are near real-time given moderate model complexities. In the case of a highly complex model (32 zones in Table 5-5), the calculation was complete in slightly over a minute.

While the models used in the experiment are abstract (Figure 5-2), they are designed to simulate the complexities in actual building designs. Identical windows within a space in reality can be abstracted to a single larger window without affecting the LEED credits calculations. Furthermore, having more windows within each space would only slow down the manual calculation (more entries to tabulate) but not the automatic one, since the view-out algorithm employs discrete sampling towards candidate windows, and can terminate once steradian thresholds are reached. The models are thus designed to focus on space adjacencies to offer view-path complexities.

The models are designed to represent single to double levels in building designs, since actual building designs typically employ typical floor layouts. In actual practice, the calculation results merely have to be multiplied to obtain overall building results. As such, typical designs rarely exceed the number of zones/windows in the models tested. Even though the LEED Credit 8.2 calculation times observe a quadratic trend, the indicative gradient and magnitude suggests the algorithms are more than sufficient to demonstrate significant time and effort savings in actual use.

6. A Scalable and Integrated Lighting Simulation Tool

This phase of the research implements a new lighting tool that drastically reduces the time and effort to prepare for lighting simulation, and complements integrated concurrent design processes for high performance buildings.

6.1. Overall Design

In this phase, a new lighting design tool is designed to reduce the time and effort required to conduct lighting simulation, and complement integrated design processes. The objectives of the new tool are:

1. Reduce the time and effort to set up and conduct lighting simulation by using interoperable information from design modeling tools.
2. Complement integrated design processes by supporting design models of varying completeness, in a format that is interoperable with tools from other disciplines in the design team. All information, including assumptions, must be consistent across all disciplines.
3. Provide ability to use consistent performance metrics and technical approaches throughout design stages, regardless of completeness of design model.
4. Provide operative information with minimum user effort.
5. Implement a first principle-based rendering technique that handles high performance building designs well, and produce simulation results within reasonable time constraints.

Functionally, the new lighting tool should import building information models from design tools, regardless of the level of completeness, and prepare simulation input files automatically. While the entire process is automatic, the user should have the option to inspect or edit all parameters. The new tool should use the new rendering technique developed earlier for simulation, and include visualization tools to analyze simulation results. Similarly, the new tool should use the automatic LEED benchmark calculators developed earlier to dynamically present LEED benchmark results. These desired features of the new lighting tool are summarized as five use cases in the simplified use case diagram below (Figure 6-1).

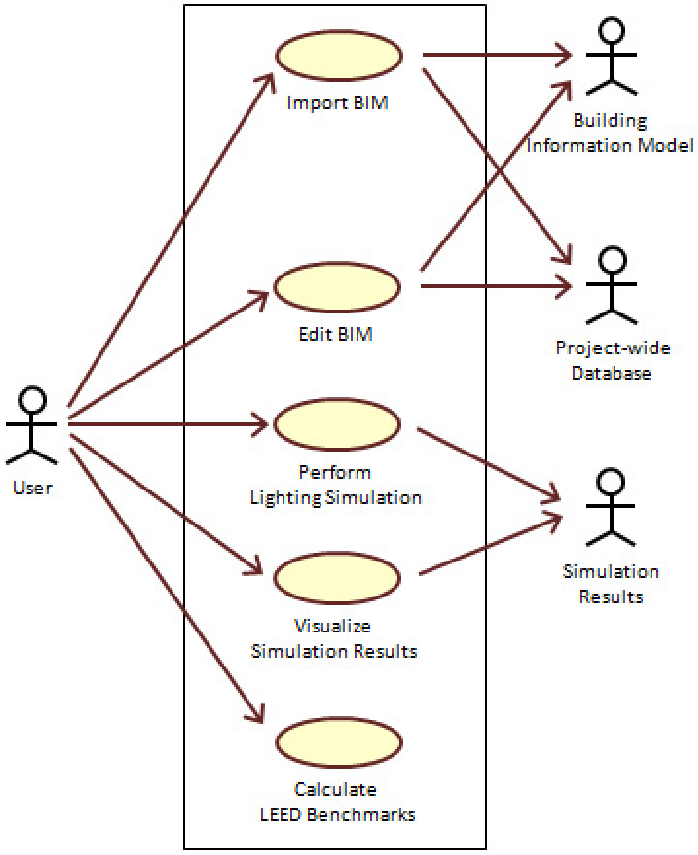


Figure 6-1 Basic Use Case Diagram of New Lighting Tool

Since the new tool should support models of varying levels of completeness, but the use of the new rendering technique and LEED calculators require complete models, this implies that the new tool performs additional work to form a complete model. A complete model is defined in this case as sufficient for lighting-domain simulation and calculations. This model is thus termed the Domain Object Model (DOM). The entire building information model, including non-lighting-domain information, as well as project-wide databases, is shared across the entire project team and constitutes the Shared Object Model (SOM). Since DOMs are proper subsets of the SOM, the SOM/DOM setup facilitates interoperability between various tools in different domains, and the information and assumptions specified in any tool can be easily checked for errors and consistency across all domains (Figure 6-2).

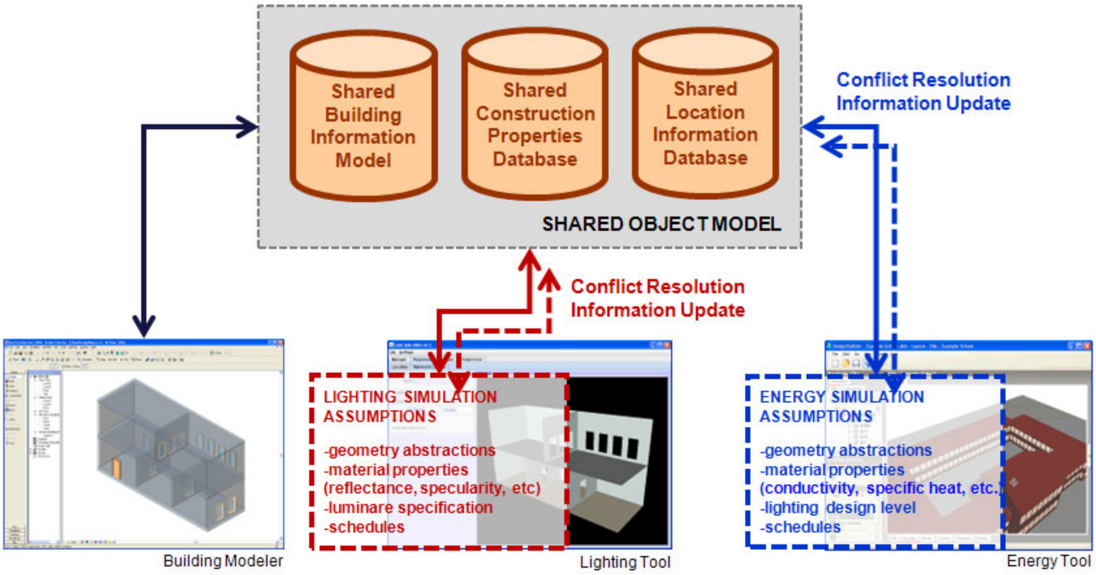

Shared Building Information Model
Shared Construction Properties Database
Shared Location Information Database
SHARED OBJECT MODEL
Conflict Resolution Information Update
Conflict Resolution Information Update
LIGHTING SIMULATION ASSUMPTIONS
-geometry abstractions
-material properties (reflectance, specularity, etc)
-luminare specification
-schedules
ENERGY SIMULATION ASSUMPTIONS
-geometry abstractions
-material properties (conductivity, specific heat, etc.)
-lighting design level
-schedules
Building Modeler
Lighting Tool
Energy Tool

Figure 6-2 Error and consistency checking across domains via SOM

Conceptually, the “Import BIM” and “Edit BIM” use cases imports lighting-related information from building information models, and subsequently form a complete DOM. To do so, the new tool fills in any missing parameters with assumptions that are documented in the external, project-wide database, which consists of a construction database and a location database. This task of forming complete DOMs, albeit automatic and not visible to the user, constitutes another use case. Likewise, the DOM maintained by the lighting tool is also a system actor since any changes to the DOM activates the “Calculate LEED Benchmarks” use case. Since the SOM is potentially accessed by different tools concurrently, a separate Change Management System should ideally be implemented to manage access and track changes to the SOM to ensure consistency and avoid data conflicts. However, the design and prototyping of such a Change Management System is beyond the scope of this research; at this point, the SOM is accessed directly by the new tool. The updated use case diagram is presented below (Figure 6-3). The actors and individual use cases are described in detail in the next two sections.

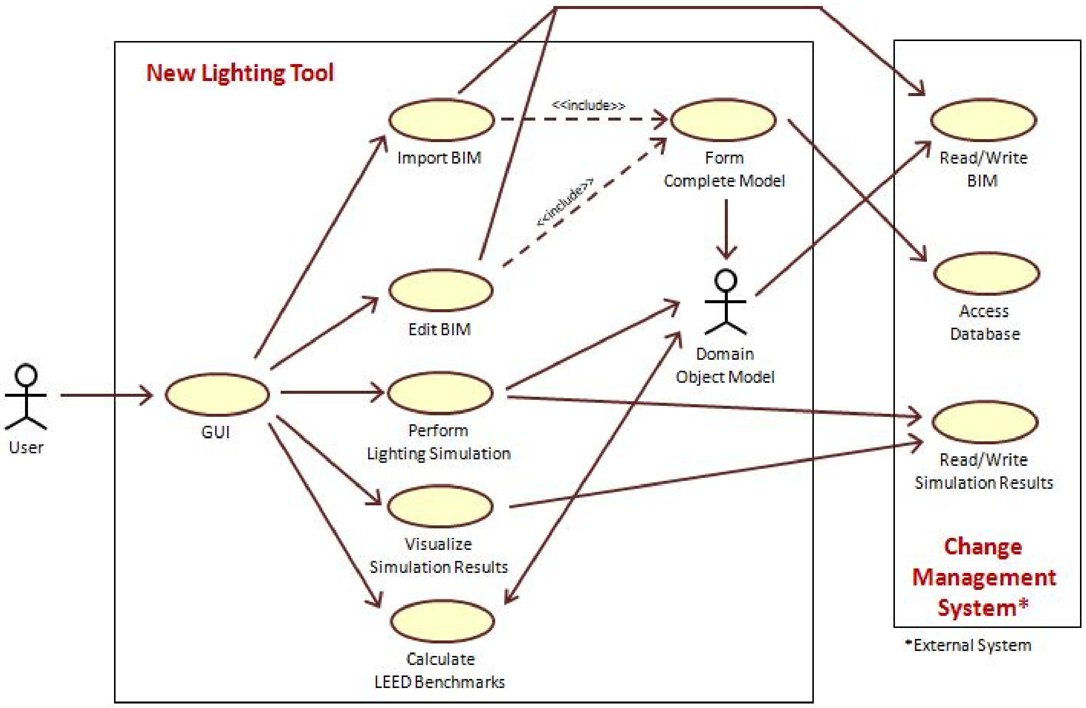


Figure 6-3 Use Case Diagram of New Lighting Tool

6.1.1. *Actors*

Actor	Description	Use-Cases
User (Human Actor)	Uses the system conduct lighting simulation activities.	Import BIM Edit BIM Perform lighting simulation Visualize simulation results Calculate LEED Benchmarks
Building Information Model (System Actor)	Information from design modeling tool in interoperable data exchange format. Part of SOM. Used by the system to achieve interoperability with other tools.	Import BIM Edit BIM
Project-wide Database (System Actor)	Location, construction, and material libraries, and assumptions in interoperable data exchange format to be shared across design team. Part of SOM. Used by the system to populate missing parameters in lighting-domain models.	Form Complete Model

Domain Object Model (DOM) (System Actor)	A complete lighting-domain model. Used by the system to perform lighting simulation and calculations. Updates in this model triggers re-calculation of LEED Benchmarks if necessary.	Form Complete Model Perform lighting simulation Calculate LEED Benchmarks
Simulation Results (System Actor)	External file containing simulation results. Used by system to store/retrieve results.	Perform lighting simulation Visualize simulation results

Table 6-1 Descriptions of Actors

6.1.2. Use-Cases

Use-Case-1 Import BIM	
Goal	Import building information models from design modeling tool
Summary	Import interoperable information from design modeling tool. **Use-Case-3 Form Compete Model** is automatically activated upon success to complete DOM for lighting simulation and calculations.
Primary Actor	User
Secondary Actor	Building Information Model (via Change Management System)
Preconditions	1. Location and Construction Databases successfully loaded by system
Triggers	1. User selects a valid file using UI 2. User saves or discard previous data using UI
Basic Course of Events	When Building Information Model is well formed 1. Building Information Model is parsed and checked to be well-formed 2. Information is populated in DOM 3. **Use-Case-3 Form Compete Model** is activated
Alternative Path	None
Exceptions Path	When Building Information Model is not well formed 1. Building Information Model is parsed and found to be not well-formed 1.1. Errors are displayed using UI 2. Information is not populated in DOM 2.1. All system data structures are cleared 3. Terminate, post conditions not implemented
Post-conditions	1. System maintains link to Building Information Model 2. DOM is well-formed, might be partially complete

Table 6-2 Use Case 1 – Import BIM

Use-Case-2 Edit BIM	
Goal	Edit building information model
Summary	User inspects and edits model attributes using UI. **Use-Case-3 Form Compete Model** is automatically activated upon success to complete DOM for lighting simulation and calculations. If LEED benchmarks have been previously calculated, **Use-Case-6 Calculate LEED Benchmarks** is automatically activated to dynamically show changes in the benchmark values.
Primary Actor	User
Secondary Actor	Building Information Model (via Change Management System)
Preconditions	1. Location and Construction Databases successfully loaded by system 2. System has link to Building Information Model 3. DOM is well-formed and complete
Triggers	1. User modifies model attribute value using UI (Editable attributes are part of location, building, construction, material, or camera objects)
Basic Course of Events	When existing attribute value is modified (location/material/view parameters) 1. Attribute field in UI updated with modified value 2. **Use-Case-3 Form Compete Model** is activated 3. System checks validity of modified value 4. DOM is updated with modified value 4.1. Attribute tagged as user-defined 5. Related UI objects updated 5.1. UI attribute field reflects "user-defined" status 5.2. If system has LEED Benchmark values, **Use-Case-6 Calculate LEED Benchmarks** is activated
Alternative Path	When new attribute value is specified (location/construction/view parameters) 1. Attribute field in UI updated with new value 2. **Use-Case-3 Form Compete Model** is activated 3. System checks validity of new value 3.1. System checks if object with specified value exists in system library and creates new library object if not so 4. DOM is updated with new value 4.1. Attribute tagged as user-defined 5. Related UI objects updated 5.1. UI attribute field reflects "user-defined" status 5.2. New object added to corresponding UI object lists 5.3. If system has LEED Benchmark values, **Use-Case-6 Calculate LEED Benchmarks** is activated

Exceptions Path	When user specified new attribute value is out-of-range, Step 2 returns valid range from SOM which Step 3 uses to determine that there is an error 3. System checks validity of new value, value is invalid 4. UI attribute field is reverted to previous value 5. Terminate, post conditions not implemented 5.1. No changes to DOM, which is still well-formed and complete 5.2. New object discarded, related UI objects not modified
Post-conditions	1. DOM is well-formed and complete 2. System includes newly created objects, related UI objects updated 3. UI option to save DOM is enabled

Table 6-3 Use Case 2 – Edit BIM

Use-Case-3 Form Complete Model	
Goal	Ensure the DOM is well-formed and complete
Summary	**Use-Case-3 Form Compete Model** is automatically activated by **Use-Case-1 Import BIM** and **Use-Case-2 Edit BIM** to populate missing parameters in DOM, such that a complete model is maintained and ready for lighting simulation and calculations.
Primary Actor	User
Secondary Actor	Location and Construction Database (via Change Management System)
Preconditions	1. Location and Construction Databases successfully loaded by system 2. System has link to Building Information Model 3. (When activated by **Use-Case-1 Import BIM**) DOM is maintained by system, well-formed but not necessarily complete 4. (When activated by **Use-Case-2 Edit BIM**) DOM is maintained by system, well-formed and complete
Triggers	1. **Use-Case-1 Import BIM** successfully completed 2. **Use-Case-2 Edit BIM** - Model attribute modified by user
Basic Course of Events	(**Use-Case-1 Import BIM**) The DOM is checked and missing attributes are populated 1. Default assumptions loaded from Location and Construction Databases 2. Preferences (default assumptions) updated in UI 3. Check and populate DOM (tagged as non-user-defined): 3.1. Location attributes 3.2. Building attributes 3.3. Construction attributes 3.4. Material attributes

	3.5. Camera attributes
Alternative Path	(**Use-Case-2 Edit BIM**) User modifies attribute, system checks if object exists in database and calculates valid range for the attribute 3. Check if modified object exists in Location/Construction Database IF ATTRIBUTES DESCRIBE EXISTING OBJECT 3.1. Retrieve object from SOM for inclusion in DOM IF ATTRIBUTES DESCRIBE NEW OBJECT 3.1. Assign new object name that does not conflict with existing DOM or SOM objects 3.2. Prepare new object for inclusion in DOM 4. Calculates valid range based on dependent attributes 5. Returns valid range for attribute
Exceptions Path	None
Post-conditions	1. DOM is well-formed and complete 2. UI option to save Building Information Model is enabled 3. If new library object is created, UI option to save project-wide Location and Construction Database is enabled

Table 6-4 Use Case 3 – Form Complete Model

Use-Case-4 Perform Lighting Simulation	
Goal	Performs lighting simulation using new rendering technique
Summary	System prepares lighting simulation inputs using information in DOM, and executes simulation using the new progressive rendering technique. Simulation results are displayed in the UI, and can be saved to the SOM to be continued later (incremental simulation). Alternatively, user can use RADIANCE for simulation instead.
Primary Actor	User
Secondary Actor	DOM, SOM
Preconditions	1. DOM is well-formed and complete 2. State of simulation obtained
Triggers	1. User requests simulation (new rendering technique or RADIANCE) via UI
Basic Course of Events	New simulation requested 1. Simulation input parsed from DOM 1.1. Simulation parameters parsed from DOM 1.2. Geometry objects parsed from DOM 2. Preprocessing of functions and maps

	2.1. Simulation maps and data structure generated from geometry objects 2.2. Light distribution functions pre-processed and cached 2.3. BRDFs pre-processed and cached 3. Lighting simulation executed
Alternative Path	Incremental simulation requested 1. Simulation input incremented from previous simulation step 2. Check that required functions and maps have been preprocessed IF FUNCTIONS AND MAPS ALREADY PREPROCESSED 2.1. Continue IF FUNCTIONS AND MAPS NOT PREPROCESSED 2.1. Simulation maps and data structure generated from geometry objects 2.2. Light distribution functions pre-processed and cached 2.3. BRDFs pre-processed and cached 3. Lighting simulation executed RADIANCE simulation requested
Exceptions Path	If simulation terminates with error 1. Terminate, post conditions not implemented
Post-conditions	1. UI option to save simulation results is enabled 2. UI updated for next step of progressive simulation

Table 6-5 Use Case 4 – Perform Lighting Simulation

Use-Case-5 Visualize Simulation Results	
Goal	Performs analysis and visualization of simulation results
Summary	System includes several tools for analyzing simulation results, including luminance and illuminance false-color renderings, contrast-ratio renderings, and a results comparator to visualize quantitative differences between two sets of simulation results
Primary Actor	User
Secondary Actor	Simulation Results
Preconditions	1. DOM is well-formed and complete
Triggers	1. User requests visualization via UI

Basic Course of Events	1. Meta-data of simulation results parsed and cached: 1.1. Name of underlying model used for simulation 1.2. Camera settings 1.3. Type of results (luminance or illuminance) 2. UI options updated according to type and dynamic range of results: 2.1. Tone-mapping 2.2. False-color scale 2.3. Normalized adaptation luminance value 2.4. Analysis options
Alternative Path	When comparing across two sets of simulation results, system checks that underlying models are related 1. Meta-data of 2nd set of results parsed and checked with 1st set 1.1. Warning displayed in UI if mismatch in underlying model or camera settings 2. UI options updated according to dynamic range of 2nd set results: 2.1. False-color scale 2.2. Normalized adaptation luminance value 2.3. Analysis options
Exceptions Path	Simulation results cannot be compared, revert to basic course of events (for analysis of single set of simulation results) 1. Meta-data of 2nd set of results parsed and checked with 1st set 1.1. Error displayed in UI if mismatch in type of results (luminance or illuminance) 2. 2nd set of results discarded 3. UI options updated according to type and dynamic range of results: 3.1. Tone-mapping 3.2. False-color scale 3.3. Normalized adaptation luminance value 3.4. Analysis options
Post-conditions	None

Table 6-6 Use Case 5 – Visualize Simulation Results

Use-Case-6 Calculate LEED Benchmarks	
Goal	Automatically calculate LEED EQ8.1 and EQ8.2 benchmarks
Summary	System automatically calculates LEED benchmarks and presents results in submittal formats. Use-case is activated by **Use-Case-2 Edit BIM** to dynamically update benchmarks when model is modified. Since
Primary Actor	User

Secondary Actor	DOM
Preconditions	1. DOM is well-formed and complete
Triggers	1. User requests calculation via UI OR 1. LEED results already exist and **Use-Case-2 Edit BIM** successfully completed
Basic Course of Events	1. Required parameters obtained from DOM 2. LEED calculation algorithms executed
Alternative Path	None
Exceptions Path	None
Post-conditions	1. LEED results cached in system

Table 6-7 Use Case 6 – Calculate LEED Benchmarks

6.1.3. *Integrating Lighting Simulation Tasks, and Supporting Integrated Design*

As presented in literature review, there is a myriad of software and tools to support the many activities involved in the building delivery process (2.1.3), of which lighting simulation is but one task. Similarly there are disparate tasks and software involved just to conduct lighting simulation (Chapter 3). This research develops new techniques and frameworks addressing primarily the latter, with the overall objective of reducing time and effort to conduct lighting simulation. Through the use-cases presented above, the new lighting tool integrates (and automates much of) the disparate tasks involved in performing lighting simulation, and the results are demonstrated below (6.3).

Although the new lighting tool does not integrate the multi-domain tasks involved in integrated design, it achieves interoperability with other tools by supporting data exchange with an external project-wide BIM-based Shared Object Model (SOM). This is a step towards facilitating concurrent, multiple domain, activities necessary in integrated design, since this modular approach allows other domain activities to proceed independently. Notably, this strategy of using interoperability to allow various simulation models to perform simulation and analysis on a consistent set of information is commonplace. Even within contemporary multi-domain analysis tools, information is passed between separate simulation modules and each (domain) simulation is performed independently; there is yet to be a single *general* simulation engine that describes all performance aspects comprehensively. Anecdotally, the developmental effort to achieve integrated simulation via independent modules operating on interoperable information is lower than formulating a single *general* simulation model.

6.2. Implementation

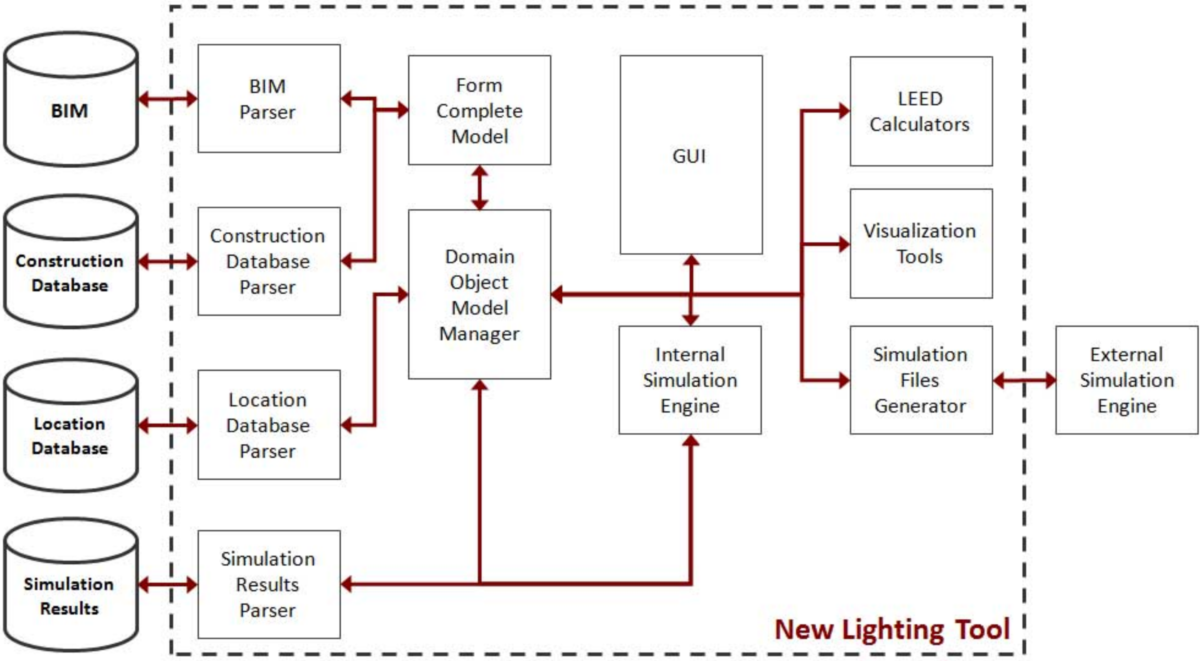


Figure 6-4 Implemented modules in new lighting tool

Schematically, the new tool is implemented as a series of modules (Figure 6-4) to achieve the desired functionalities as described by the use-cases above (6.1.2). The modules are presented in the section below (6.2.2).

For ease of prototyping, the new tool is developed in object-orientated fashion using Java 2 Platform Standard Edition 5.0, Java3D, and Java API for XML Processing (JAXP). Since Building Information Models are parametric and object-based, it follows that object-orientated programming (OOP) would lend easily to the task of developing the new tool. In addition, OOP is also more suited to implementing the various modules of the new tool (as described in the figure above); each module is independent and well encapsulated, and can be implemented and tested in OOP fashion. Additionally, the high-level nature of the Java programming language, platform-independence, as well as the availability of extensive Java Class Libraries and application programming interfaces (APIs) in Java, Java3D, and JAXP, especially for user interfaces (UI) and multi-threading, enables relatively fast prototyping.

While the code executes slower in Java (as compared to lower-level languages), and there is less control over memory management, empirical tests in this research have shown that the all implemented features execute within desired time-constraints, and the 1.4 gigabytes of memory available to the Java Virtual Machine (as configured in this research) is sufficient for even complex architectural scenes. A 2.66 GHz single CPU workstation with 2 GB of RAM using Windows XP OS, representative of the reasonable computing resources available in small architectural practices, is used for all the work (including simulation) in this research.

6.2.1. gbXML as Data Exchange Format

As presented earlier (1.2.1), prevalent design modeling tools currently support both IFC and gbXML schemas as data exchange formats, and both formats are extensible to represent information necessary for lighting simulation (2.1.3). While there are significant differences between the two, including comprehensiveness, efficiency, robustness, redundancies, and portability, the debate over which schema is better as a data exchange format for Building Information Models is beyond the scope of this research. Since the goal of this research is to demonstrate effort and time-cost reductions by implementing interoperable information models and datasets, rather than espousing the technical benefits of particular exchange formats, both schemas can potentially meet the needs and objectives of this research.

Within this context, the choice of data format for prototyping the new lighting tool in this research is only premised upon capability to demonstrate the concept of an integrated lighting simulation tool, ease of development and implementation. This include extensibility to capture information comprehensively, ease of implementing extensions, prevalence in industry, and ability to demonstrate interoperability between design modeling tools, the new lighting tool, and another domain simulation tool.

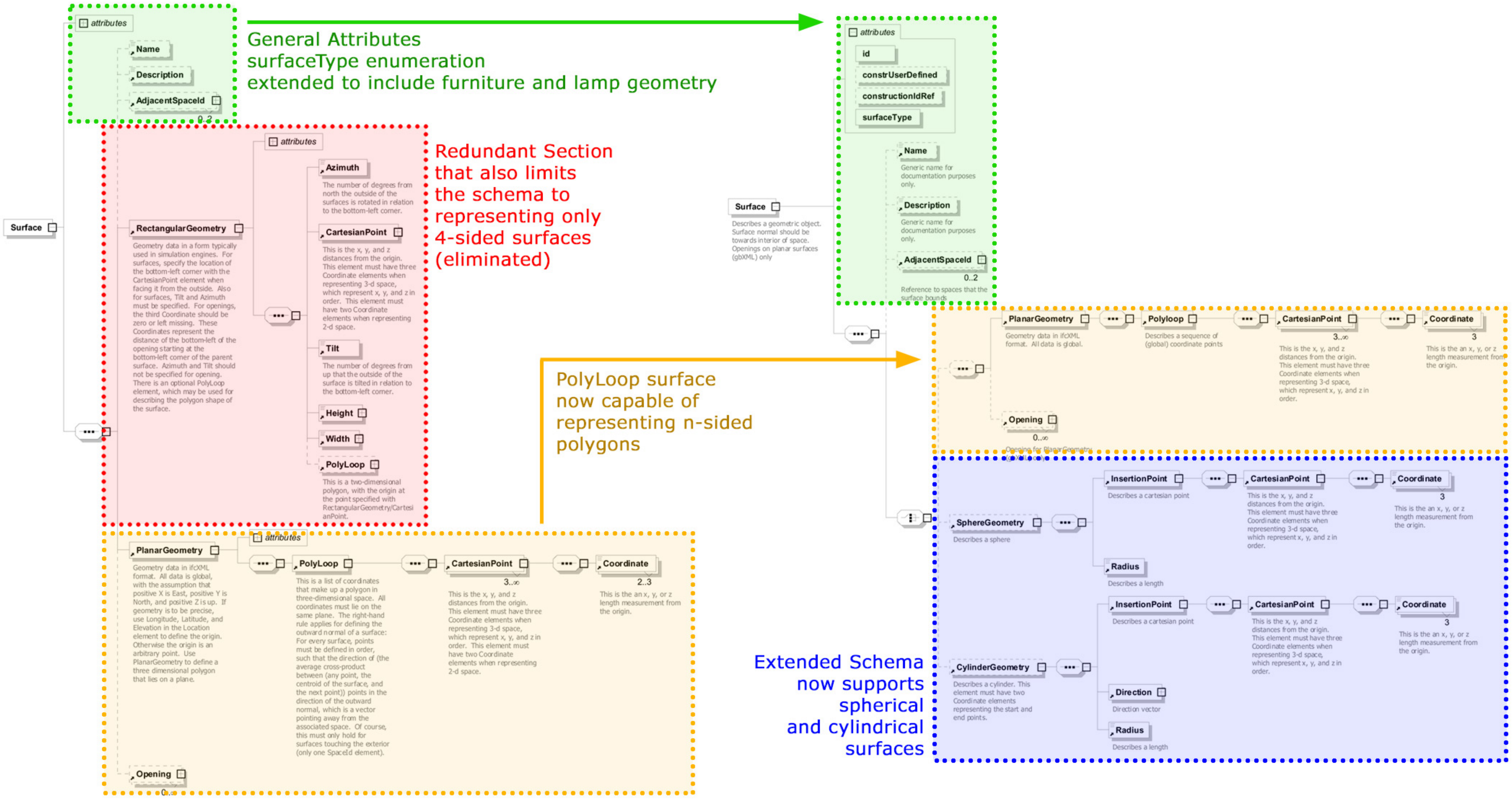
As discussed earlier (2.1.3), both formats can represent parametric building information, and are extensible to represent *comprehensive* building information, including that necessary for lighting simulation. The gbXML data format is selected for use given the ease of legibility and extension when developing prototypes (Dong, Lam, Huang, & Dobbs, 2007). While extending gbXML necessitates a change to the overall schema (by introducing new object attributes and properties via XML tags), this is relatively easier to develop as compared to extending general purpose entities with defined base entity and relationship objects in the IFC schema. In addition, the availability of Java API for XML Processing (JAXP), allows ease of development and implementation of XML-based features within the new tool.

To support integrated design processes, the data exchange format should ideally be supported by all other domain tools; the BIM should be concurrently accessible by multiple domain tools to facilitate an *integrated* performance evaluation. As mentioned earlier (2.1.3 & 6.1.3), there is yet to be a single *integrated* tool or model, and the use of multiple tools, at least in the near future, is inevitable. Since the motivation for this research (lighting simulation) is premised upon energy use, it follows that the selected data exchange format should be supported by state-of-the-art building energy simulation tools, such as EnergyPlus, to demonstrate multi-domain support. As noted earlier (1.2.1), gbXML can be translated into EnergyPlus input formats by *middleware* such as Ecotect and GreenBuildingStudio.

	IFC	**gbXML**
Modeling Language	EXPRESS	XML
Public (Open source)	Yes	Yes
Potential to represent all types of building information	Yes	Yes
Easily Extensible	Moderate	Yes
Supported by existing Java API	No	Yes
Supported by Modeling Tool (Revit)	Yes	Yes
Supported by Energy Tool (EnergyPlus)	Yes	Yes

Table 6-8 Considerations for implementing data exchange format

A model schema that is comprehensive enough to include all necessary information and semantically compatible with the diverse domain views, yet lightweight enough for efficient query and use, is developed by extending the gbXML schema (GeoPraxis Inc., 2003) to reduce redundancy and include lighting information (Figure 6-5). This XML-based schema is used to construct a holistic BIM that can be implemented across the design team as a Shared Object Model (SOM). The SOM can then be can be parsed into several lightweight domain specific Domain Object Models (DOM) by different tools, such as by the new lighting tool developed in this research, or EnergyPlus (via Green Building Studio).



General Attributes
surfaceType enumeration
extended to include furniture and lamp geometry

Redundant Section
that also limits
the schema to
representing only
4-sided surfaces
(eliminated)

PolyLoop surface
now capable of
representing n-sided
polygons

Extended Schema
now supports
spherical
and cylindrical
surfaces

Surface
attributes
Name
Description
AdjacentSpaceId
0..2
attributes
RectangularGeometry
Geometry data in a form typically used in simulation engines. For surfaces, specify the location of the bottom-left corner with the CartesianPoint element when facing it from the outside. Also for surfaces, Tilt and Azimuth must be specified. For openings, the third Coordinate should be zero or left missing. These Coordinates represent the distance of the bottom-left of the opening starting at the bottom-left corner of the parent surface. Azimuth and Tilt should not be specified for opening. There is an optional PolyLoop element, which may be used for describing the polygon shape of the surface.
Azimuth
The number of degrees from north the outside of the surfaces is rotated in relation to the bottom-left corner.
CartesianPoint
This is the x, y, and z distances from the origin. This element must have three Coordinate elements when representing 3-d space, which represent x, y, and z in order. This element must have two Coordinate elements when representing 2-d space.
Tilt
The number of degrees from up that the outside of the surface is tilted in relation to the bottom-left corner.
Height
Width
PolyLoop
This is a two-dimensional polygon, with the origin at the point specified with RectangularGeometry/CartesianPoint.
attributes
PlanarGeometry
Geometry data in ifcXML format. All data is global, with the assumption that positive X is East, positive Y is North, and positive Z is up. If geometry is to be precise, use Longitude, Latitude, and Elevation in the Location element to define the origin. Otherwise the origin is an arbitrary point. Use PlanarGeometry to define a three dimensional polygon that lies on a plane.
PolyLoop
This is a list of coordinates that make up a polygon in three-dimensional space. All coordinates must lie on the same plane. The right-hand rule applies for defining the outward normal of a surface: For every surface, points must be defined in order, such that the direction of (the average cross-product between (any point, the centroid of the surface, and the next point)) points in the direction of the outward normal, which is a vector pointing away from the associated space. Of course, this must only hold for surfaces touching the exterior (only one SpaceId element).
CartesianPoint
3..∞
This is the x, y, and z distances from the origin. This element must have three Coordinate elements when representing 3-d space, which represent x, y, and z in order. This element must have two Coordinate elements when representing 2-d space.
Coordinate
2..3
This is the an x, y, or z length measurement from the origin.
Opening
0..∞
Surface
Describes a geometric object. Surface normal should be towards interior of space. Openings on planar surfaces (gbXML) only
attributes
id
constrUserDefined
constructionIdRef
surfaceType
Name
Generic name for documentation purposes only.
Description
Generic name for documentation purposes only.
AdjacentSpaceId
0..2
Reference to spaces that the surface bounds
PlanarGeometry
Geometry data in ifcXML format. All data is global.
Polyloop
Describes a sequence of (global) coordinate points
CartesianPoint
3..∞
This is the x, y, and z distances from the origin. This element must have three Coordinate elements when representing 3-d space, which represent x, y, and z in order.
Coordinate
3
This is the an x, y, or z length measurement from the origin.
Opening
0..∞
Opening for PlanarGeometry (gbXML) only
SphereGeometry
Describes a sphere
InsertionPoint
Describes a cartesian point
CartesianPoint
This is the x, y, and z distances from the origin. This element must have three Coordinate elements when representing 3-d space, which represent x, y, and z in order.
Coordinate
3
This is the an x, y, or z length measurement from the origin.
Radius
Describes a length
CylinderGeometry
Describes a cylinder. This element must have two Coordinate elements representing the start and end points.
InsertionPoint
Describes a cartesian point
CartesianPoint
This is the x, y, and z distances from the origin. This element must have three Coordinate elements when representing 3-d space, which represent x, y, and z in order.
Coordinate
3
This is the an x, y, or z length measurement from the origin.
Direction
Direction vector
Radius
Describes a length

Figure 6-5 gbXML Surface element (left) and extended schema (right).

6.2.2. Modules

Referring to Figure 6-3 and Figure 6-4, the various modules and actors are classified into 3 categories; as pertaining to 1) the new lighting tool, 2) the external change management system, and 3) the project-wide shared object model (SOM) (Table 6-9). The new lighting tool features described earlier in the 6 use-cases (plus GUI use-case, which is simply the interface between user and the 6 use-cases), and the DOM, are achieved by 8 corresponding modules. Additionally, a *RADIANCE Input Files Generator* module is implemented to allow comparison of simulation results between the new simulation engine and RADIANCE.

	Object in Figure 6-3 **Use-Case Diagram**	**Module in Figure 6-4** **Implemented Modules**
New Lighting Tool	Use Case - GUI	Module – GUI
	Use Case 1 – Import BIM	Module – Import BIM
	Use Case 2 – Edit BIM	Module – Edit BIM
	Use Case 3 – Form Complete Model	Module – Form Complete Model
	Use Case 4 – Perform Lighting Simulation	Module – Internal Simulation Engine
	Use Case 5 – Visualize Simulation Result	Module – Visualization Tools
	Use Case 6 – Calculate LEED Benchmarks	Module – LEED Calculators
	Actor – Domain Object Model	Module – Domain Object Model Manager
	Link to RADIANCE	Module – RADIANCE Input Files Generator
External Change Management System	Use Case – Read/Write BIM	Module – BIM Parser
	Use Case – Access Database	Module – Construction Database Parser
		Module – Location Database Parser
	Use Case – Read/Write Simulation Results	Module – Simulation Results Parser
(Project-wide) Shared Object Model	Actor – Building Information Model	Database – BIM
	Actor – Construction Database	Database – Construction
	Actor – Location Database	Database – Location
	Actor – Simulation Results	Database – Simulation results

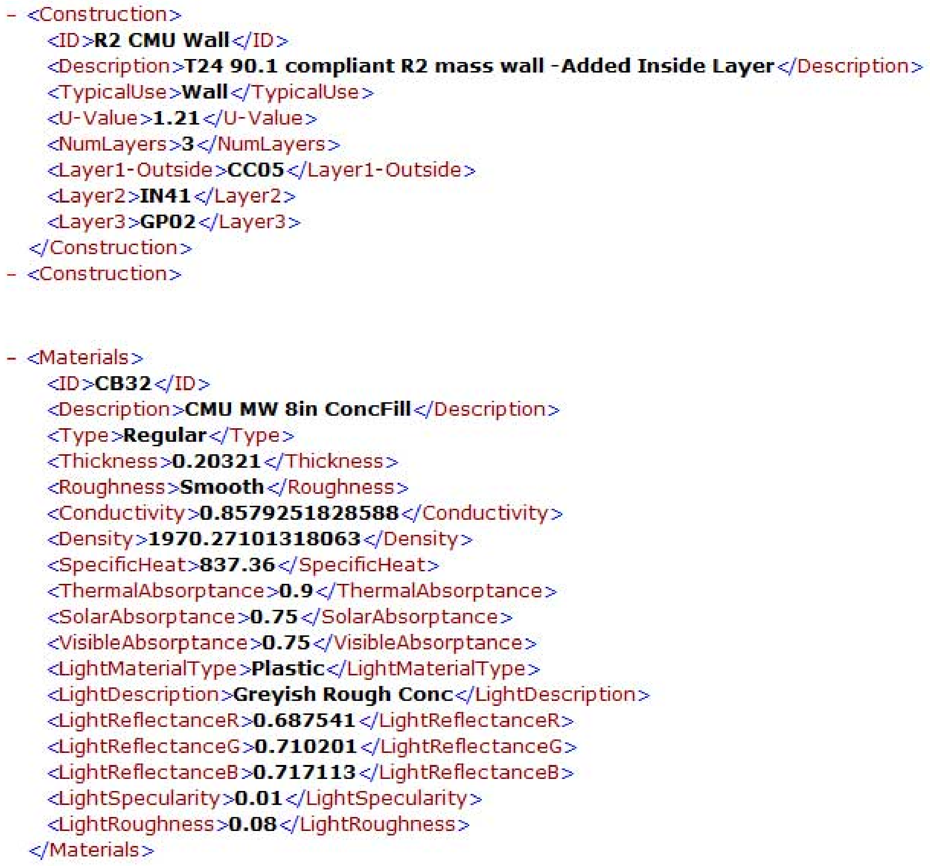
Table 6-9 Correspondence between objects in use-case diagram and implemented modules

While the development of data repositories and external change management systems as discussed earlier (2.1.3 & 6.1) are beyond the scope of this project, a demonstrative project-wide SOM is implemented via 4 separate databases, and 4 corresponding modules are implemented to allow the new lighting tool to access the 4 databases. The 4 databases in the SOM cover the building information model (BIM), construction and location databases, and simulation results. While the BIM includes project-specific construction and location information, the databases serve as libraries for ease of specification during design, and avoids erroneous specifications that conflicts with other domains.

6.2.2.1. Shared Object Model (SOM) Databases

As discussed earlier (6.2.1), the current gbXML schema (version 0.37) is extended to cover all necessary data elements to represent information, such as sky descriptions, n-sided polygons, detailed material reflectance properties and bidirectional reflectance distribution functions (BRDF), as required for lighting simulation. This extended schema is used to describe holistic building information, forming the BIM database which is shared across the project team.

Following the principles and benefits of utilizing a SOM, similar project-wide, application-independent datasets of construction and location information (for United States) are also developed. The same extended XML-based schema is used to organize this information to ensure portability and ease of parsing.

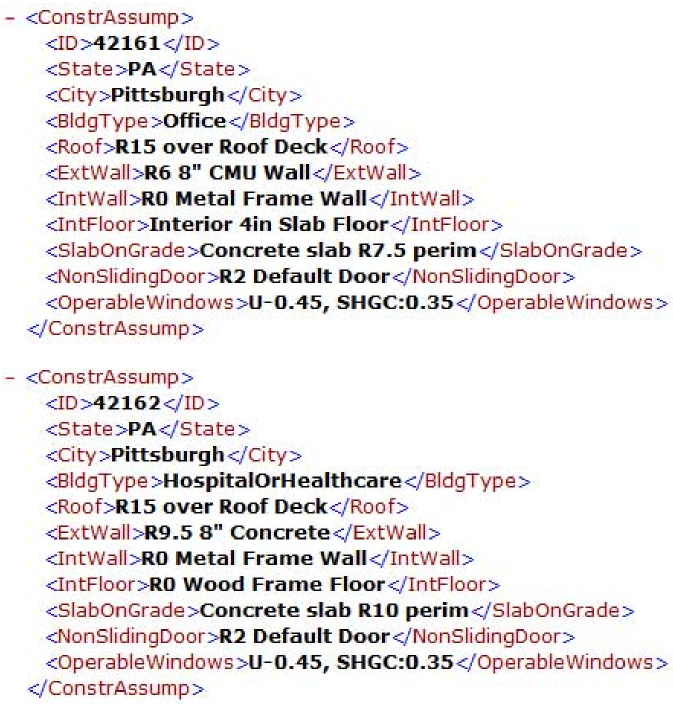


```
- <Construction>
    <ID>R2 CMU Wall</ID>
    <Description>T24 90.1 compliant R2 mass wall -Added Inside Layer</Description>
    <TypicalUse>Wall</TypicalUse>
    <U-Value>1.21</U-Value>
    <NumLayers>3</NumLayers>
    <Layer1-Outside>CC05</Layer1-Outside>
    <Layer2>IN41</Layer2>
    <Layer3>GP02</Layer3>
  </Construction>
- <Construction>


- <Materials>
    <ID>CB32</ID>
    <Description>CMU MW 8in ConcFill</Description>
    <Type>Regular</Type>
    <Thickness>0.20321</Thickness>
    <Roughness>Smooth</Roughness>
    <Conductivity>0.8579251828588</Conductivity>
    <Density>1970.27101318063</Density>
    <SpecificHeat>837.36</SpecificHeat>
    <ThermalAbsorptance>0.9</ThermalAbsorptance>
    <SolarAbsorptance>0.75</SolarAbsorptance>
    <VisibleAbsorptance>0.75</VisibleAbsorptance>
    <LightMaterialType>Plastic</LightMaterialType>
    <LightDescription>Greyish Rough Conc</LightDescription>
    <LightReflectanceR>0.687541</LightReflectanceR>
    <LightReflectanceG>0.710201</LightReflectanceG>
    <LightReflectanceB>0.717113</LightReflectanceB>
    <LightSpecularity>0.01</LightSpecularity>
    <LightRoughness>0.08</LightRoughness>
  </Materials>
```

Figure 6-6 Example of construction and material elements in construction database

Given the fact that the building industry typically employs a relatively limited variety of standard practices, the preparation of this shared data set is relatively straightforward. The construction database includes typical construction types and material definitions. To demonstrate interoperability, the material definitions include both lighting and thermal properties (Figure 6-6). The construction database also contains typical construction practices for different building types in various cities across the United States (Figure 6-7). This information is used by the *Form Complete Model* module to ensure completeness of models, as described later (6.2.2.6).

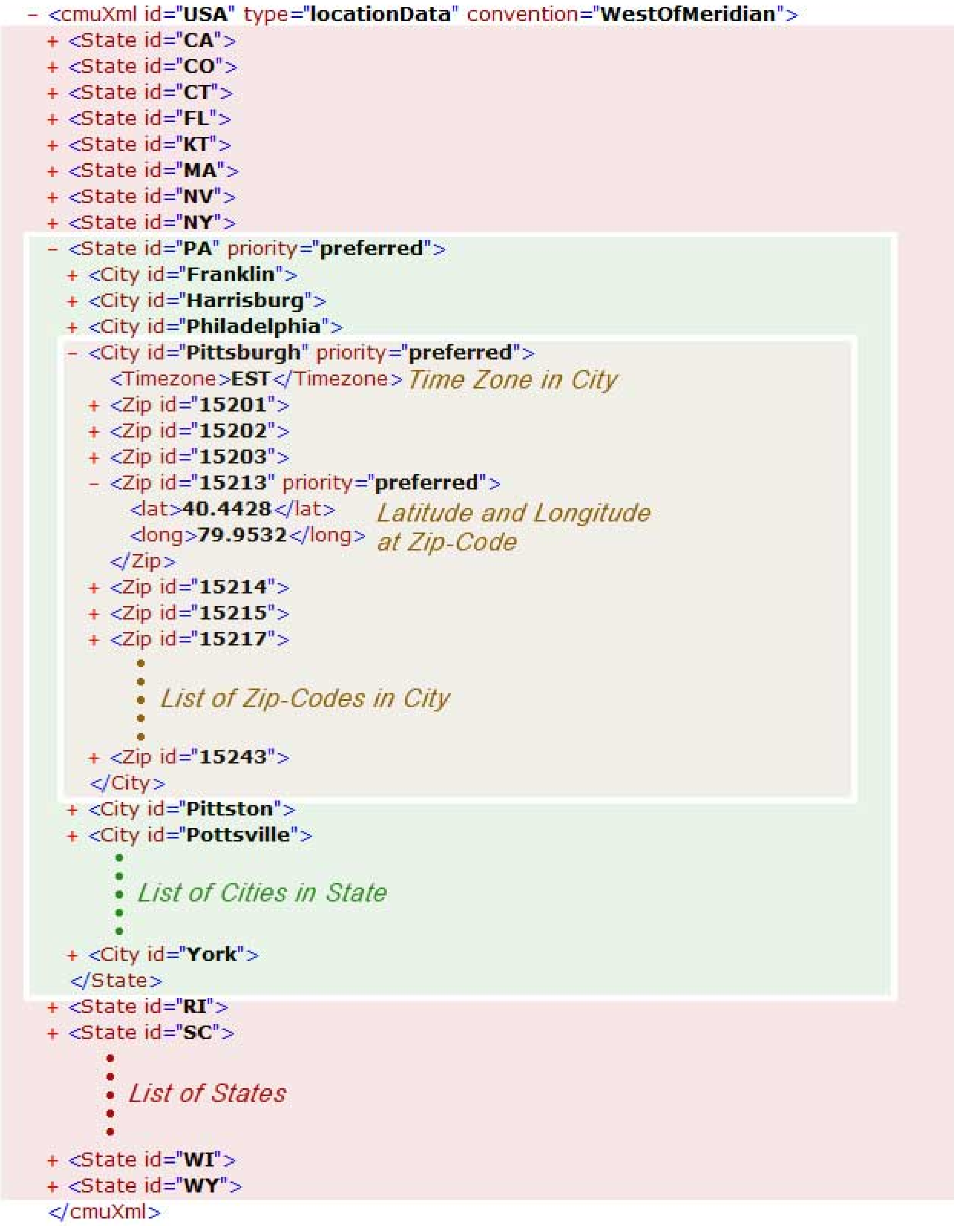


```
- <ConstrAssump>
    <ID>42161</ID>
    <State>PA</State>
    <City>Pittsburgh</City>
    <BldgType>Office</BldgType>
    <Roof>R15 over Roof Deck</Roof>
    <ExtWall>R6 8" CMU Wall</ExtWall>
    <IntWall>R0 Metal Frame Wall</IntWall>
    <IntFloor>Interior 4in Slab Floor</IntFloor>
    <SlabOnGrade>Concrete slab R7.5 perim</SlabOnGrade>
    <NonSlidingDoor>R2 Default Door</NonSlidingDoor>
    <OperableWindows>U-0.45, SHGC:0.35</OperableWindows>
  </ConstrAssump>
- <ConstrAssump>
    <ID>42162</ID>
    <State>PA</State>
    <City>Pittsburgh</City>
    <BldgType>HospitalOrHealthcare</BldgType>
    <Roof>R15 over Roof Deck</Roof>
    <ExtWall>R9.5 8" Concrete</ExtWall>
    <IntWall>R0 Metal Frame Wall</IntWall>
    <IntFloor>R0 Wood Frame Floor</IntFloor>
    <SlabOnGrade>Concrete slab R10 perim</SlabOnGrade>
    <NonSlidingDoor>R2 Default Door</NonSlidingDoor>
    <OperableWindows>U-0.45, SHGC:0.35</OperableWindows>
  </ConstrAssump>
```

Figure 6-7 Example of typical construction practices for different building types in the same city

The location database is a collection of latitude, longitude, and time-zone data for locations in the United States, cross-indexed with city names and zip-codes (Figure 6-8). This information is used by the *Form Complete Model* module, as described later (6.2.2.6), to ensure that the BIM specifies location information in terms of latitude and longitude. Note that the location database is not a weather file. The location database ensures consistent latitude and longitude information, which is then used to determine the appropriate selections in various domain simulations, such as generating sun positions for lighting simulation, or selecting the correct weather file in energy simulation.

The simulation results database contains point radiance or irradiance values in the RGBE format (Ward G. , 1991). The RGBE format is a compact high dynamic range image (HDRI) format used by RADIANCE, and subsequently there are many post-processing and analysis tools available for this format.



```
- <cmuXml id="USA" type="locationData" convention="WestOfMeridian">
  + <State id="CA">
  + <State id="CO">
  + <State id="CT">
  + <State id="FL">
  + <State id="KT">
  + <State id="MA">
  + <State id="NV">
  + <State id="NY">
  - <State id="PA" priority="preferred">
    + <City id="Franklin">
    + <City id="Harrisburg">
    + <City id="Philadelphia">
    - <City id="Pittsburgh" priority="preferred">
        <Timezone>EST</Timezone>
      + <Zip id="15201">
      + <Zip id="15202">
      + <Zip id="15203">
      - <Zip id="15213" priority="preferred">
          <lat>40.4428</lat>
          <long>79.9532</long>
        </Zip>
      + <Zip id="15214">
      + <Zip id="15215">
      + <Zip id="15217">

      + <Zip id="15243">
      </City>
    + <City id="Pittston">
    + <City id="Pottsville">

    + <City id="York">
    </State>
  + <State id="RI">
  + <State id="SC">

  + <State id="WI">
  + <State id="WY">
  </cmuXml>
```

Figure 6-8 Latitude, longitude, and time-zone data for locations in United States

6.2.2.2. External Change Management System Modules

In lieu of supporting some external data repository or change management system, 4 parser modules are implemented for the new tool to access the BIM, construction, location, and simulation results databases. While this framework links the new lighting tool directly to the SOM (Figure 6-4), conceptually these 4 parser modules represent some independent change management system that could be used to manage the SOM (Figure 6-3 & Figure 6-9). All tools, including the lighting tool, would then access the SOM via the independent system to ensure consistency in assumptions made in each domain. Specific user definition of variables and parameters can also be validated against these datasets automatically without the need for user intervention. While the choice and implementation of a change management system is beyond the scope of this research, the modular framework allows such future development.

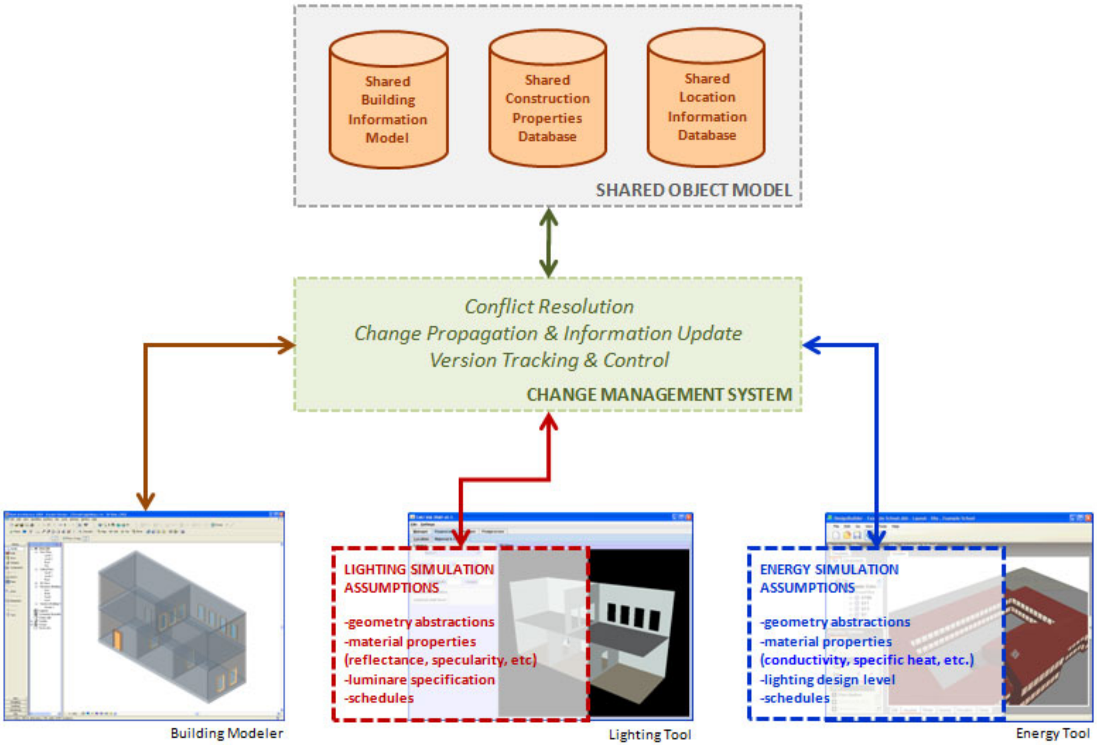
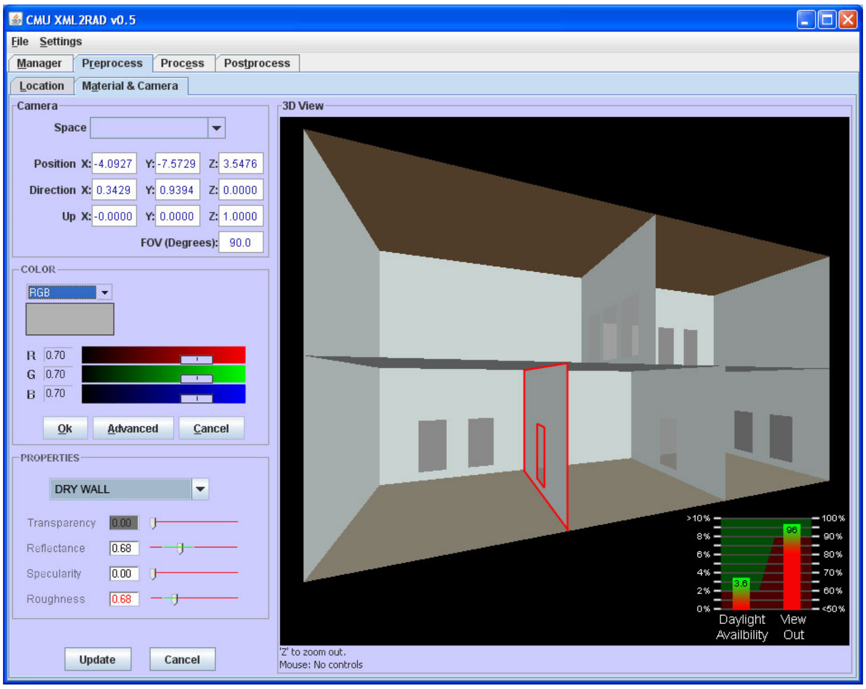

Shared Building Information Model
Shared Construction Properties Database
Shared Location Information Database
SHARED OBJECT MODEL
Conflict Resolution
Change Propagation & Information Update
Version Tracking & Control
CHANGE MANAGEMENT SYSTEM
LIGHTING SIMULATION ASSUMPTIONS
-geometry abstractions
-material properties
(reflectance, specularity, etc)
-luminare specification
-schedules
ENERGY SIMULATION ASSUMPTIONS
-geometry abstractions
-material properties
(conductivity, specific heat, etc.)
-lighting design level
-schedules
Building Modeler
Lighting Tool
Energy Tool

Figure 6-9 External Change Management System between SOM and domain tools

As described above, the first three databases (BIM, construction, and location) are implemented in the XML-based schema, and the results database in the RGBE format. Correspondingly, the first 3 parser modules uses the Java API for XML Processing (JAXP) to read and write to the databases, while the last parser module implements algorithms to compress and decompress RGBE data.

6.2.2.3. GUI Module


CMU XML2RAD v0.5

Space:

Position X: -4.0927 Y: -7.5729 Z: 3.5476

Direction X: 0.3429 Y: 0.9394 Z: 0.0000

Up X: -0.0000 Y: 0.0000 Z: 1.0000

FOV (Degrees): 90.0

COLOR: RGB

R 0.70

G 0.70

B 0.70

PROPERTIES: DRY WALL

Transparency 0.00

Reflectance 0.68

Specularity 0.00

Roughness 0.68

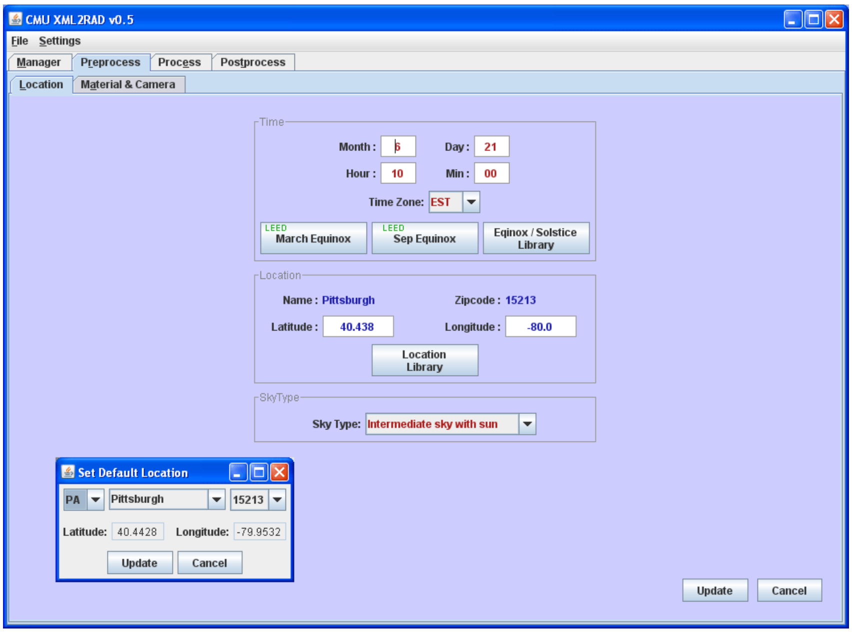
'Z' to zoom out.

Mouse: No controls

Figure 6-10 Overall GUI design of new lighting tool

The graphical user interface (GUI) is implemented using Swing toolkits, which is part of the Java Foundation Classes (JFC), and Java3D. The design (Figure 6-10) of the GUI reinforces the workflow of simulations via a series of tabbed displays corresponding to the stages of simulation work process, and the display of parameters is flexible to reflect the changing levels of detail (LOD) along different stages of design. The GUI employs color coding to cognitively distinguish between user specified values and those that are populated automatically by querying the datasets. Similarly, the recommended ranges of values for various parameters are presented in the same manner (Figure 6-11).

To facilitate faster cognition of the context of various attributes, the new tool uses an interactive 3D model viewer to let users inspect and edit the information. The tabulation of the LEED benchmarks (6.2.2.10) are also dynamically linked to this viewer; changes made to the model results in an instantaneous update of the LEED evaluations. The tabulation format of these benchmarks is also consistent with submission requirements. This contributes to the reduction of common time consuming manual activities.



Month : 6 Day : 21
Hour : 10 Min : 00
Time Zone: EST

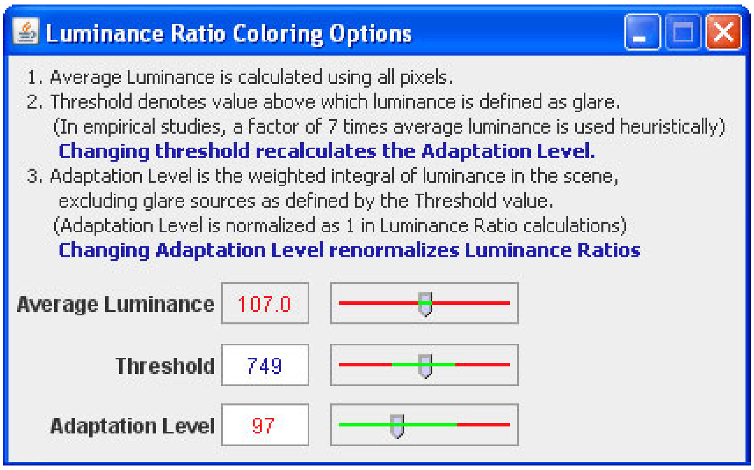
Name : Pittsburgh Zipcode : 15213
Latitude : 40.438 Longitude : -80.0

Sky Type: Intermediate sky with sun

PA Pittsburgh 15213
Latitude: 40.4428 Longitude: -79.9532

Figure 6-11 Automatic completion of location information as parsed from incomplete BIM.
Smaller window shows optional user selection of preferred location,
automatic populated values in red, BIM information in blue.

To improve the ease of use of the tool, help menus and documentation within the GUI makes explicit some of the tacit knowledge in conducting simulation. Together with the mentioned recommended value ranges and color-coding, the inspection of parameters and use of appropriate values is made easier; there is no need for additional research or depend on user expertise. Another objective of the documentation is to avoid the inappropriate use of metrics by highlighting the underlying methodologies in a succinct manner (Figure 6-12).

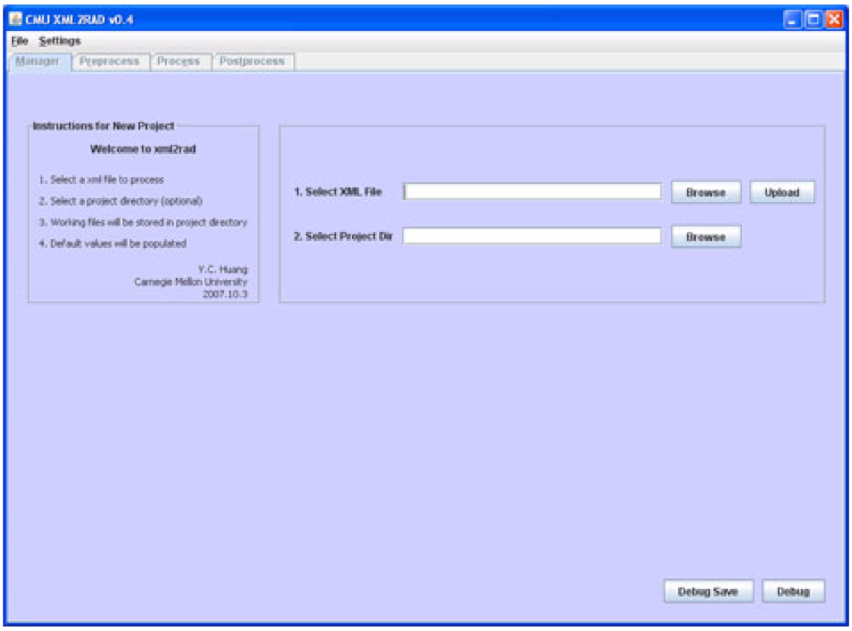


1. Average Luminance is calculated using all pixels.
2. Threshold denotes value above which luminance is defined as glare.
(In empirical studies, a factor of 7 times average luminance is used heuristically)
Changing threshold recalculates the Adaptation Level.
3. Adaptation Level is the weighted integral of luminance in the scene,
excluding glare sources as defined by the Threshold value.
(Adaptation Level is normalized as 1 in Luminance Ratio calculations)
Changing Adaptation Level renormalizes Luminance Ratios

Average Luminance 107.0
Threshold 749
Adaptation Level 97

Figure 6-12 Luminance Ratio GUI

6.2.2.4. Import BIM Module



Instructions for New Project

Welcome to xml2rad

1. Select a xml file to process
2. Select a project directory (optional)
3. Working files will be stored in project directory
4. Default values will be populated

Y.C. Huang
Carnegie Mellon University
2007.10.3

1. Select XML File

2. Select Project Dir

Figure 6-13 Import BIM GUI

In keeping with the overall goal of reducing time and effort, the Import BIM module is developed to require minimal user intervention. The user only has to select a file via the GUI, and the BIM is ready for lighting simulation or calculations; all necessary processing are automatic. The module parses information from the XML-based BIM using JAXP, and checks that it is well-formed. The module then populates the DOM (6.2.2.7), and activates the Form Complete Model Module (6.2.2.6) to ensure that the DOM is complete.

```
ImportBIM() {
    if ( BIMParser.parseBIM() ) {       use BIM Parser Module to check BIM is well-formed
        DOMManager.populateDOM()        activates  DOM Manager Module to create DOM
        DOMComplete.complete()          use FormCompleteModel Module to complete DOM
                                        automatically
    }
    updateGUI()                         updates the GUI
}
```

Figure 6-14 Pseudocode for importing BIM

Via the BIM parser, the new lighting tool translates the SOM, as generated by a design modelling tool (Revit Architecture), to automatically to form a DOM suitable for use by lighting simulation. There is now a seamless sharing and reuse of building information between the design modelling tool (Revit Architecture), the new lighting tool, and to demonstrate integrated design, an energy modelling tool[27]. The preparation of the lighting model from a single SOM shared across different teams and domains in integrated design processes also eliminates the need for error- and consistency checking. This automatic semantic translation essentially eliminates the single largest obstacle and time-cost of using lighting simulation as indicated by the mentioned research and surveys (learning and using multiple tools, repetitive manual re-modeling, error- and consistency checking).

6.2.2.5. Edit BIM Module

While the imported BIM is complete and ready for lighting simulation, the user can inspect and edit all parameters, such as location and sky-type (Figure 6-11), construction types, material properties, or camera attributes (Figure 6-10). The link to the project-wide databases in the SOM allows user-defined changes to be validated against these datasets automatically without the need for user intervention, and ensures consistency across domains. Upon confirmation of user-defined changes, the LEED calculation results (6.2.2.10) are dynamically updated.

```
EditBIM() {
  if ( ! drop down menu selected ) {        objects in drop down menus are valid objects
                                            parsed from SOM databases

    if ( ! SOM.has( modified ) ){           check SOM if object with modified attributes
                                            exists

      displayWarning()                      prompt user modified attribute is not in SOM
                                            database
      if ( userCancels() ){
        modified = original                 revert to original attribute if user cancels change
      } else {
        SOM.add( modified )                 add modified object to SOM
      }
    }
  }
  DOMManager.update( modified )             updates DOM Manager Module with new object
  LEED.update()                             updates results in LEED Module
  updateGUI()                               updates the GUI
}
```

Figure 6-15 Pseudocode for editing BIM

6.2.2.6. Form Complete Model Module

The new tool maintains a well-formed and complete DOM at all times, which is accessed by other modules in the tool. The module maintains a set of default values for all parameters

[27] gbXML-based BIMs are used by GreenBuildingStudio (Autodesk, 2009) to conduct energy simulation

necessary for lighting simulation, and populates the DOM automatically with these values when missing attributes are encountered. Indicatively, the DOM is structured into 7 sections:

1. Location attributes
2. Building attributes
3. Construction attributes
4. Material attributes
5. Surface attributes
6. Lamp attributes
7. Camera attributes

The databases in the SOM contains a hierarchy of default object types and attribute values to be used consistently across the project team in the absence of specifications in the BIM. For example, in the absence of location information, the default location (Figure 6-8) is at latitude 40.4428, longitude 79.9532 (Pittsburgh, Pennsylvania, zip-code 15213). In the location database, each state has a default city, and each city has a default locale specified by zip-code. The module is thus able to automatically specify missing latitude/longitude data regardless of the type of partial, or missing, location description. Conversely, in the case when only latitude and longitude data is available in the BIM, the module performs a nearest neighbor search to determine the city and state of the location, and appends it to the BIM for legibility.

The structure of default object types is described as *hierarchical* because the choice of objects from the SOM database is dependent on higher-level information. The order of importance corresponds with the list above. Default building attributes, such as building use-type (office, retail, school, etc.), are dependent on location information. Default surface attributes, such as material definition, are dependent on the default construction method (which specifies layers of materials) for that specific surface-type (roof, wall, floor, window, etc.), which in turn is dependent on building-type and location.

Taking an example where no information except for surface geometry is present in a BIM, the module would thus use the default location (Pittsburgh, Pennsylvania), which determines the default building-type (office), and consequently a set of default construction methods, which includes specifying 4-inch concrete slabs for interior floors. Since this construction specifies concrete as the outermost material layer, the concrete material object is also appended to the

BIM. The concrete material object contains lighting-related attributes such as red, green, and blue reflectance values. In the event that the surface described in this *sparse*-BIM is an interior floor, the module has now formed a *complete* BIM that is ready for lighting simulation (the surface has been populated with default values necessary for lighting simulation).

The same method is applied for all sections of attributes. The SOM databases contain default building-types, construction-types for building elements, and material properties. In the event of missing objects in the BIM, the default objects are used. In the case when partial information is encountered, the module searches the databases and retrieves the missing information or default child-elements. All data specified by the module are tagged as *default values* and presented in a different color in the GUI for legibility (Figure 6-10 & Figure 6-11). In the case of camera attributes, the module ensures that at least 1 camera is defined. In case where there is no camera, the module inserts a standard camera, with height and angle-of-view similar to human vision, in the geometric center of the first occupied space specified in the BIM.

```
completeDOM () {

  object = DOM.getLocation()              Step 1: Check  Location objects
  default = SOM.getDefaultLocation()

  if( object == null ){
    DOM.setLocation( default )            no location in DOM, use default location
  } else {
    if( ! object.isComplete() ){
      complete( object )                  calls complete() to fill in missing location data in
    }                                     DOM
  }

  object = DOM.getBuilding()              Step 2: Check  Building objects
  default = SOM.getDefaultBuilding( DOM.getLocation() )
                                          default building attributes are dependent on locale
  if( object == null ){
    DOM.setBuilding( default )            no building attributes in DOM, use default values
  } else {
    if( ! object.isComplete() ){
      complete( object )                  calls complete() to fill in missing building data in
    }                                     DOM
  }

                                          Step 3: Check Construction and Material objects
  for ( all construction and material objects in DOM ){
    complete( object )                    calls complete() to fill in missing building data in
  }                                       DOM
```

```
                                          Step 4: Check surfaces for construction links
  defaultTypes = SOM.getConstrTypes( DOM.location, DOM.buildingType)
                                          default construction methods for different surface-
                                          types are dependent on locale and building-type
  for ( all building surface objects in DOM ){

    object = surface.getConstruction()
    type = DOMSurface.type
    default = defaultTypes.type           specific default construction method for surface

    if ( object == null ){
      DOMSurface.setConstr( default )     no construction linked to surface, use default
      DOM.add( default )                  append construction and material objects to DOM
    }                                     no such state as partial construction link
  }

                                          Step 5: Check Lamp objects
  for ( all lamp objects in DOM ){
    complete( object )                    calls complete() to fill in missing lamp data in
  }                                       DOM


  if ( no camera object in DOM ){         Step 6: Check Camera objects
    DOM.insert( generateCamera() )        generates and insert camera object in DOM
  } else {
    for ( all camera objects in DOM ){
      complete( object )                  calls complete() to fill in missing camera data in
    }                                     DOM
  }
}


                                          Helper function to complete DOM objects
complete( object ){
  default = SOM.search( object )          search SOM for nearest match given object with
                                          partial attributes values

  for( all attributes in object ){        complete missing  attributes with default values
    if( attribute == null )               when necessary
      object.attribute.setValue( default.attribute )
  }

  for( all child objects in object ){     complete missing child objects with default
    if( object.child == null ){           values when   necessary
      object.child.setObject( default.child )
    } else {
      if( ! object.child.isComplete() ){
        complete( object.child )          recursive call to complete attributes and sub-
                                          objects of existing chid object when necessary
    }
  }
}
```

Figure 6-16 Pseudocode for automatically completing the DOM

6.2.2.7. Domain Object Model Manager Module

As illustrated in the use-case diagram (Figure 6-3), the new tool maintains a single Domain Object Model (DOM), for use by all other use-cases. This DOM is managed by the DOM Manager Module (Figure 6-4), which utilizes the gbXML schema as discussed earlier (Chapter 6.2.1). The module is implemented using JAXP, which allows relative ease of maintaining the DOM in the XML format.

While the DOM is initially empty when the new tool is launched, user access to all other tool features except Import BIM is blocked since the tool is designed to process BIMs. This is reflected in all feature tabs in the GUI being grey-out and non-selectable (Figure 6-13, only the BIM file-select menu is available upon tool launch). Once the user selects a BIM file, the SOM is imported and processed into a complete and well-formed DOM, and passed to the DOM Manager Module for subsequent use by all other modules in the tool (as discussed in 6.2.2.4). Once the DOM is ready, the module updates the GUI to enable all features by enabling all tabs in the GUI. In this manner, the DOM Manager Module maintains a complete and well-formed DOM at all times, and ensures fail-safe.

6.2.2.8. Internal Simulation Engine Module

The new tool implements the new simulation technique as described earlier (Chapter 4.3) to solve the three-component rendering equation (Eq. 4-4) consistently.

$$
\begin{aligned}
&L_o(x,\vec{\omega}) = L_e(x,\vec{\omega}) + L_r(x,\vec{\omega}) \\
\Rightarrow\ &L_o(x,\vec{\omega}) = L_e(x,\vec{\omega}) + \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_i(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \\
\Rightarrow\ &L_o(x,\vec{\omega}) = L_e(x,\vec{\omega}) + \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_{i,l}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \\
&\qquad + \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_{i,d}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}' \\
&\qquad + \int_\Omega f_r(x,\vec{\omega}',\vec{\omega})L_{i,c}(x,\vec{\omega}')(\vec{n}\cdot\vec{\omega}')d\vec{\omega}'
\end{aligned}
$$

Eq. 6-1

Where $L_{i,l}\left(x,\vec{\omega}'\right)$ is direct illumination from light sources (**LD** paths), $L_{i,d}\left(x,\vec{\omega}'\right)$ is indirect illumination from light sources that has been reflected diffusely at least once (**LD[S|D]*D** and

L[S|D]*DD paths), and $L_{i,c}\left(x,\vec{\omega}'\right)$ is caustics – indirect illumination from the light sources via specular reflection or transmission (**LS+D** paths).

Similar to classic photon mapping (4.2), the new technique adopts a two-pass approach, a ray-tracing pre-process to more efficiently evaluate direct and specular ray-paths, followed by a photon tracing and gathering process to more accurately account for highly reflected indirect ray-paths. The difference however, is that the new technique focuses on the *components* of all contributing points rather than the type of *path* of the points. The implemented pre-process uses recursive ray-tracing to identify the surface intersection point associated with each pixel, as well as all mirrored, transmitted, and refracted surface intersection points that contribute towards that point. The emitted radiance and direct illuminance at all the points is determined and cached, together with intersection information including surface normal ($\vec{n}$) and view angles ($\vec{\omega}$), to facilitate efficient evaluation of indirect illumination components (diffuse and caustics) via the photon map in the later photon gathering process.

```
firstPass(){                              First-pass cache list of points (Le and Ld) contributing
                                          towards intersection point associated with pixel radiance
  for ( all pixels ){
    generate new Ray object from camera and pixel
    intersections = trace( ray, 0, 1 )     Calls trace() to recursively trace Ray,
                                            initial depth = 0, power = 1 (100%)

    for ( intersections ){                 Cache emitted and direct illumination for list of
                                            contributing intersections

      if( intersect.type == lamp ){        Cache radiance of intersected lamp for given
        intersect.Le =                     intersection attributes (location & view direction)
            lamp.getRadiance( intersect )
      }
      Intersect.Ld =                       Cache radiance due to direct illumination from
        intersect.directIllum( allLamps )   all lamps and material BRDF
    }
    cache intersections list for each pixel
  }
}


trace( ray, depth, power ){               Recursive ray-tracing to identify list of specular reflected
                                           opaque surface  intersections (with power factors)

  stop if ( depth > limit                  Terminate recursion if too deep or power
                || power < limit2)         becomes insignificant

  get nearest ray-surface intersection     Get first ray-surface intersection
  mat = intersection.getMaterial()         Retrieve material from appropriate side
  if( mat.type == mirror ){                 Special case, only mirrored points, factored by
    generate mirror Ray at intersection     material reflectance, contribute to  pixel
    power *= mat.reflectance                irradiance.
    return trace( mirrorRay, depth+1,       Recursive call to trace() to get list of
                             power )        mirrored surface intersections.
  }
  if( mat.type == dielectric ){             Special case, reflected and refracted points,
    generate reflected Ray                  factored by material reflectance and
    generate refracted Ray                  transmittance contribute to pixel irradiance.
    power1 = power * mat.reflectance                 transmittance = reflection + refraction
    power2 = power * ( mat.transmittance             transmittance +absorption = 1
                     - mat.reflectance )
    list = trace( reflectRay, depth+1,      Recursive trace() to obtain reflected points
                             power1 )
    list.add(trace( refractRay, depth+1,    Recursive trace() to add refracted points
                             power2 ))
    return list
  }
  intersection.power = power                Cache cumulative power factor to scale radiance
                                            contribution to pixel later in photonGather()
  return intersection                       Intersection is opaque material ( Ward Model),
                                             indirect irradiance evaluated in photon-pass
}
```

Figure 6-17 Pseudocode for new simulation technique, ray-tracing pre-process

The second-pass consists of emitting photons from light sources (photon emission, Figure 4-7) and estimating the irradiance at points as identified in the ray-tracing pass using the new simulation technique as described earlier (4.3.1), which includes using the (Ward model-based) three-component rendering equation (Eq. 4-4) consistently, reflected caustics, surface-based photon maps, and adaptive-sampling area estimates.

To achieve scalability, the second-pass can be executed iteratively to progressively increase the accuracy of the simulation results. In each iteration, additional photons are emitted from light sources and traced in the scene, and the previous irradiance estimates updated. To manage the increasing computation costs and resources associated with more photons, the number of emitted photons is increased quadratically while the number of samples in radiance estimates is increased linearly (4.3.2).

To improve the performance of the iterative technique in typical high performance building contexts (highly reflected light paths), a power-based priority queue is also implemented as a form of importance-sampling for photon emission in each iterative step (4.3.3). The priority queue (photonPowerHeap in Figure 6-18) is implemented as a binary heap of photon references (with photon-surface intersection information for ease of emitting new photons via Russian roulette from the intersection) sorted by the degree of preference $P(x)$(Eq. 3-8). A photon-relations map (phoRelMap in Figure 6-18) implemented as a tree that stores the relationships between all photons[28]. In the power-prioritized pass, the indexes of preferred photons (parent photons) are retrieved from the heap. The relevant node in the photon-relations map is retrieved and the power of all children branches halved. The additional photon (with half the parent photon radiant flux before BRDF attenuation) and subsequent children (as determined by Russian Roulette), is then appended as a branch beneath the parent photon node.

```
2ndPass( step, intersections ){        Second pass iterated (steps) to progressively improve
                                       accuracy for intersections list identified in pre-process

  numPho = Math.power(2,step)                 Quadratic increase in number of emitted photons
  numSamp = step                              Linear increase in number of irradiance estimate
                                              samples
  data.prep()                                 Pre-process and cache material BRDF ,lamp
                                              distributions, and data structures

  emitPhoton( step, numPho )                  Photon emission from light sources and power-
                                              based priority queue

  balPhoton()                                 Balance photon maps (kd-trees) for efficiency

  gatherPhoton( intersections, numSamp )      Evaluate irradiance at required points
}


emitPhoton(  step, numPho  ){          Emit photons and maintain power-based priority queue

  power = 1 / ( 179*numPho )                  Radiometry convention 179 W/lumen

  for ( allLamps ) {                          Step 1:  Emit photons from light sources
    for ( lamp.lumens * numPho ){             Determine number of photons to emit
                                              for each lamp
      Photon spawn = lamp.spawn(index++)      Emit indexed photon from lamp
      spawn.setPower(power)                   Calibrate power of photon

      photonTrace(spawn, 0, true, null)       Trace photon in scene, initial depth = 0,
```

[28] In Russian-roulette photon tracing, photons intersecting opaque surfaces are stored in photon maps (kd-trees), then reflected (determined by sampling the surface BRDF) and traced to subsequent surface intersections. Path segments are modeled independently; *parent* photons are emitted and ray-traced to the next opaque intersection and stored in the photon map, a new *child* photon is emitted to represent the reflected photon. All photon paths in a scene can be captured in a map; photons emitted from light sources are independent top-level nodes, and subsequent light paths are represented as branches of parent-child nodes representing each reflection in the overall light path. While original photon tracing results in a series of linear branches, the implemented power-based importance-sampling causes forking in the branches.

```
                                               pathAllSpecular = true, photonRelation = null
  }
}

for( numPho ){                                 Step 2: Trace some photons from power-
                                               based priority queue (heap)

  degPref = phoPowerHeap.queryRootValue() Query P(x) value of root node in heap
  retrieved = photonPowerHeap.getRoot()  Retrieve and remove root node from heap
  mat = retrieved.mat                    Material at node photon- surface intersection
  power = retrieved.photon.power         Photon radiometric power

  node = phoRelMap.get(retrieved.index)  Retrieve node in photon-relations map using
                                         index of photon
  numChild = node.numChild()
  scale = numChild / ( numChild+1 )      Scale factor for existing child nodes
  scale2 = 1 / ( numChild+1 )            Scale factor for new child photon

  prob = random[0,1]                     Russian roulette to determine type of refection
                                         for new child photon

  if( prob < mat.pd ){                   Random variable is less than parent node
                                          intersection material diffuse reflectance pd.


    Photon spawn =                       New photon is diffuse reflection. Material BRDF
          mat.spawnDiffused(             and parent surface-intersection orientation used
          index++, retrieved.attrib )    to generate new indexed child photon

    spawn.setPower( scale2 * power )     Scale power of new child nodes
    node.scaleChildPower( scale )        Scale power of existing child nodes
    phoPowerHeap.add( retrieved, degPref/2 )   Heap node reinserted with half original
                                               P(x) value.
    photonTrace(spawn, 1, false, node)   Trace photon in scene, depth > 0 since reflected
                                          at least once, pathAllSpecular = false since
                                          diffuse reflection, photonRelation node retrieved
                                         above

  } else if( prob < (retrieved.mat.pd + retrieved.mat.ps) ){
                                         Material diffuse reflectance pd and specularity ps

    Photon spawn =                       New photon is specular reflection. Material BRDF
          mat.spawnSpecular(             and parent surface-intersection orientation used
          index++, retrieved.attrib )    to generate new indexed child photon

    spawn.setPower( scale2 * power )     Scale power of new child nodes
    node.scaleChildPower( scale )        Scale power of existing child nodes
    phoPowerHeap.add( retrieved, degPref/2 )   Heap node reinserted with half original
                                               P(x) value.
    spec = retrieved.spec                Inherit path history of parent photon
    photonTrace(spawn, 1, spec, node)    Trace photon in scene, depth > 0 since reflected
                                          at least once, pathAllSpecular inherited from
                                         parent since specular reflection, photonRelation
                                          node retrieved above
  } else {
                                         Absorption – no new child photon emitted.
```

```
                                          Reinsert original heap node.
      phoPowerHeap.add( retrieved, degPref )
    }
  }
}


photonTrace( photon, depth,          Recursive photon tracing, terminating by depth.
            allSpec, relNode ){      Stores results to photon maps, power-based priority
                                     queue , and photon-relations map

  stop if ( depth > limit                 Terminate recursion if too deep or power
                  || power < limit2)      becomes insignificant

  generate ray from photon                Photon paths follows ray optics
  get nearest ray-surface intersection    Get first ray-surface intersection
  mat = intersect.getMaterial()           Retrieve material from appropriate side

  photon.setOrigin( intersection point )  Update parent photon position
  pho = new Photon( photon, index++ )     Create new child photon based on parent photon
  newNode = new photon-relations node     Add child node to photon-relationship map later
  prob = random[0,1]                      Russian roulette to determine type of refection

  if( mat.type == mirror ){               Special case, photon either mirrored or absorbed
    if( prob<mat.ps ){
      find mirror direction               Photon mirrored.
      pho.setDirection( mirrDir )         Update child photon
      degPref = pho.power * mat.ps        Calculate child photon P(x)
    } else end                            Photon absorbed, do nothing
    depth -= 1                            Prevent depth increment below, light mirrored
                                          directly from light sources (depth=0) are
                                           evaluated in ray-tracing pre-process
  }
  else if( mat.type == dielectric){       Special case, photon reflected, refracted or
                                           absorbed.
    if( prob<mat.ps ){                    Photon reflected (for glass, ps = reflection)
      find reflection direction
      pho.setDirection( reflectDir )      Update child photon
      degPref = pho.power * mat.ps        Calculate child photon P(x)
    }
    else if( prob<mat.transmittance ){    Photon refracted.
      find refracted direction            Transmittance = reflection + refraction
      pho.setDirection( refractDir )      Update child photon
      degPref = pho.power *               Calculate child photon P(x)
            (mat.transmittance - mat.ps)  transmittance = reflection + refraction
    } else end                            Photon absorbed, do nothing
    depth -= 1                            Prevent depth increment below, light transmitted
                                          via dielectrics directly from light sources (depth
                                          =0) are evaluated in ray-tracing pre-process
  }
  else {                                  Intersection is opaque material, photon diffuse or
                                          specular reflection, or absorbed
    if( depth!=0 ){                       Depth 0 photons represent direct illumination,
                                          and are  thus not stored in photon maps
      if( allSpec ){                      Photon path all specular reflections  so far (LS*)
        add photon to caustics map        Parent photon added to caustic map
      } else {                            Photon path at least 1 diffuse reflection (L?*D?*)
        add photon to global map          Parent photon added to global photon map
      }
    }
    if( prob<mat.pd ){                    Photon diffuse reflection

      allSpec = false                     Child photon path now contains at least 1
                                          diffuse reflection
      find new diffuse direction          Random sample cached material BRDF to obtain
                                          diffuse reflection direction, given surface normal
      pho.setDirection( diffDir )         Update child photon
      degPref = pho.power * mat.pd        Calculate child photon P(x)
    }
    else if( prob<(mat.pd+mat.ps) ){      Photon specular reflection
      find new specular direction         Random sample cached material BRDF to obtain
                                          specular reflection direction, given parent
                                          photon direction and intersection surface normal
      pho.setDirection( specDir )         Update child photon
      degPref = pho.power * mat.ps        Calculate child photon P(x)
    } else end                            Photon absorbed, do nothing
  }
                                          Re-factored tasks below. Absorbed-photon cases
                                          terminate before this point
  if( relNode!= null){                    Update photon-relationship map
    relNode.addChild( newNode )           Add child node to parent node reference
  }else{
    phoRelMaps.add( newNode )             Add new node to map as top-level node
    relNode = newNode
  }
  photonPowerHeap.add( photon, degPref )  Add parent photon to heap
  if( pho!=null )
      photonTrace( pho, depth+1,          Trace child photon recursively
                   allSpec, relNode )
}


gatherPhoton (intersections, numSamp){    Photon gathering to evaluate overall radiance
                                          of a point given list of contributing intersections

  reset ans                               Overall radiance is sum of 4-component radiance
                                          from all contributing intersection points
  for ( all intersections ){              L = Le + Ld + Li + Lc
    if( intersect is lamp ){
      ans += intersect.Le                 Step1:   Le scaled and cached in pre-process
                                          trace()
    }
    ans += intersect.Ld                   Step2:   Ld scaled and cached in pre-process
                                          trace()
    mat = intersection material
    if( mat.ps>0 ){
      reset specRad                       Step3a: Find specular component of Li
      numSamp2 = numSamp *                Determine number of samples to evaluate
            ( mat.ps/(mat.ps+mat.pd) )    radiance Li due to specular-part of BRDF
      for( numSamp2 ){
        specRay =                         Random sample material BRDF for specular
            mat.sampSpec( intersect )     component direction, given view direction and
```

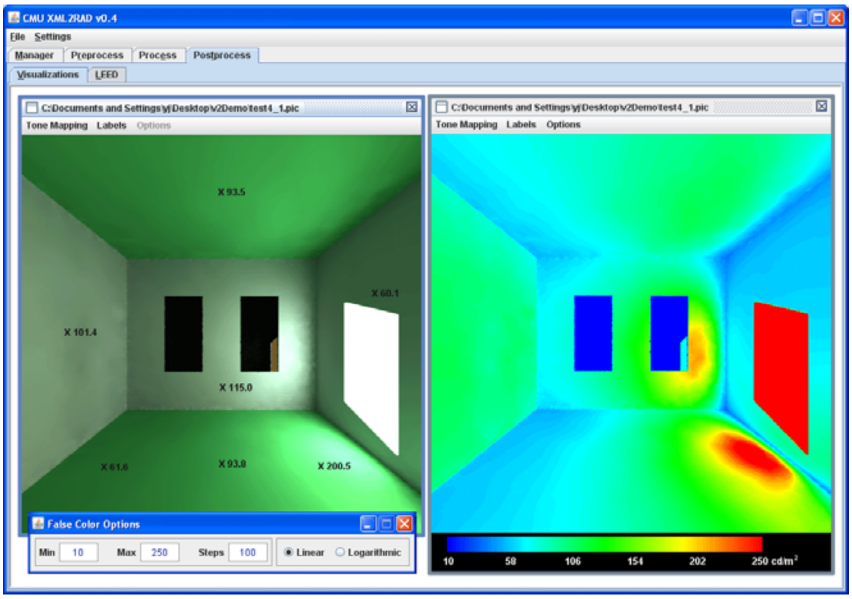
```
                                          create new Ray object
        specSampList =                    Recursively trace sample ray to obtain first
            trace( specRay, 0, 1 )        opaque surface-intersection or list of
                                          intersections if specularly transported
        for( specSampList ){
          specRad += estRadiance          Estimate radiance at specular sampled point
                  ( specSamp, intersect ) towards intersection, factored by power factor
        }                                 of sampled point
      }
      ans += specRad/numSamp2             Adjust specular radiance by number of samples
    }                                     and add to overall radiance

    reset diffRad                         Step3b: Find diffuse component of Li
    numSamp3 = numSamp *                  Determine number of samples to evaluate
            ( mat.pd/(mat.ps+mat.pd) )    radiance Li due to diffuse-part of BRDF
    for( numSamp3 ){
      diffRay =                           Random sample material BRDF for diffuse
            mat.sampDiff( intersect )     component direction, given view direction and
                                          create new Ray object
      diffSampList =                      Recursively trace sample ray to obtain first
            trace( diffRay, 0, 1 )        opaque surface-intersection or list of
                                          intersections if specularly transported
      for( diffSampList ){
          diffRad += estRadiance          Estimate radiance at diffuse sampled point
                  ( diffSamp, intersect ) towards intersection, factored by power factor
      }                                   of sampled point
    }
    ans += diffRad/numSamp3               Adjust diffuse radiance by number of samples
                                          and add to overall radiance
    ans += estCaustics( intersect )       Step4:   Lc estimated from caustics map
  }
  return ans
}
```

Figure 6-18 Pseudocode for new simulation technique, photon tracing and gathering.

6.2.2.9. Visualization Module

The implemented 2-pass simulation engine produces physically accurate radiance values for the scenes as decimal values in luminance units of candelas per square meter (cd/m^2) for all pixels in a bitmap image of the scene as determined by the camera view[29]. In order to effectively visualize the numerical results, the visualization module provides functionality to process and present the data graphically. To maximize the relevance of the tool, the simulation results are encoded in the RGBE format (Ward G. , 1991). The RGBE format is a compact high dynamic range image (HDRI) format used by RADIANCE, prevalent in lighting simulation use, and subsequently there are many post-processing and analysis tools available for this format.

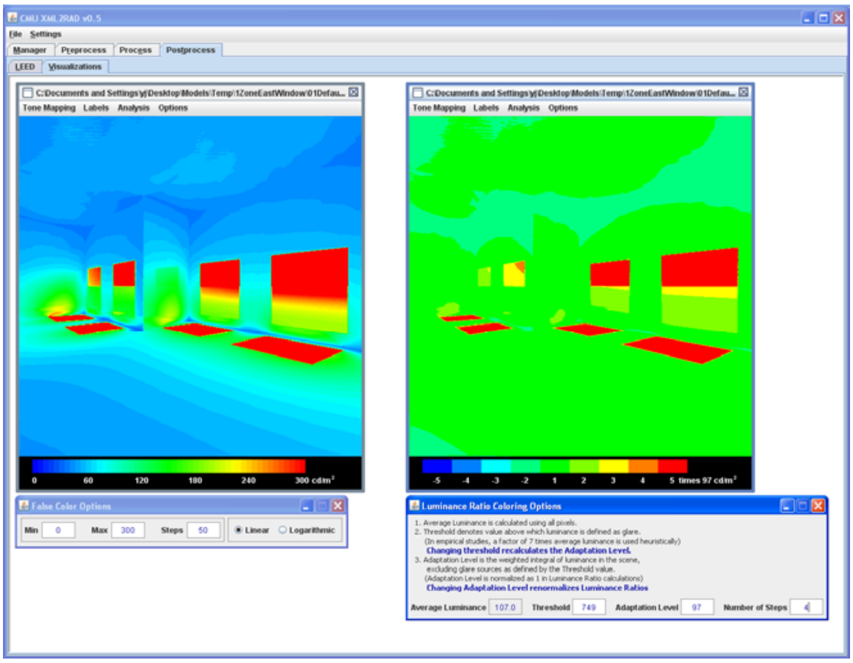

CMU XML 2RAD v0.4
File Settings
Manager Preprocess Process Postprocess
Visualizations LFED
C:\Documents and Settings\yj\Desktop\v2Demo\test4_1.pic
Tone Mapping Labels Options
X 93.5
X 101.4
X 60.1
X 115.0
X 61.6
X 93.8
X 200.5
C:\Documents and Settings\yj\Desktop\v2Demo\test4_1.pic
Tone Mapping Labels Options
False Color Options
10
58
106
154
202
250 cd/m²

Min 10 Max 250 Steps 100 ● Linear ○ Logarithmic

Figure 6-19 Visualization Module

The visualization module is implemented within the same tabbed GUI for consistency and ease of use (Figure 6-19), and includes features pertinent to the presentation of data such that it facilities design decisions (*operative information*). While the research on data cognition is beyond the scope of this research, the palette of features selected in implemented are based on the typical information needs of lighting design. This includes tone mapping of high dynamic range images for visual inspection (Larson, Rushmeier, & Piatko, 1997), iso-contour false-color representation with user-editable ranges, user-editable normalized luminance ratio analysis, and a comparator for pairs of RGBE images.

[29] Of user defined resolution, as calculated in the 1st pass of the internal simulation engine module.


CMU XML 2RAD v0.5
File Settings
Manager Preprocess Process Postprocess
LEED Visualizations
C:\Documents and Settings\yj\Desktop\Models\Temp\1ZoneEastWindow\01Defau...
Tone Mapping Labels Analysis Options
0 60 120 180 240 300 cd/m²
C:\Documents and Settings\yj\Desktop\Models\Temp\1ZoneEastWindow\01Defau...
Tone Mapping Labels Analysis Options
-5 -4 -3 -2 1 2 3 4 5 times 97 cd/m²
False Color Options
Luminance Ratio Coloring Options

Min 0 Max 300 Steps 50 ● Linear ○ Logarithmic

1. Average Luminance is calculated using all pixels.
2. Threshold denotes value above which luminance is defined as glare.
(In empirical studies, a factor of 7 times average luminance is used heuristically)
Changing threshold recalculates the Adaptation Level.
3. Adaptation Level is the weighted integral of luminance in the scene,
excluding glare sources as defined by the Threshold value.
(Adaptation Level is normalized as 1 in Luminance Ratio calculations)
Changing Adaptation Level renormalizes Luminance Ratios

Average Luminance 107.0 Threshold 749 Adaptation Level 97 Number of Steps 4

Figure 6-20 False-color (left) and luminance ratio (right) analyses
Corresponding feature options GUI below each image

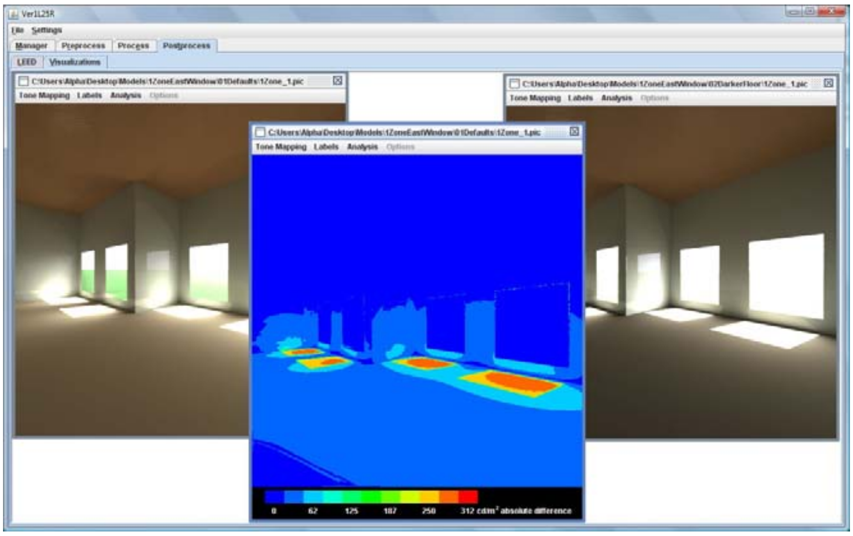

Ver1L25R
File Settings
Manager Preprocess Process Postprocess
LEED Visualizations
C:\Users\Alpha\Desktop\Models\1ZoneEastWindow\01Defaults\1Zone_1.pic
Tone Mapping Labels Analysis Options
C:\Users\Alpha\Desktop\Models\1ZoneEastWindow\02DarkerFloor\1Zone_1.pic
Tone Mapping Labels Analysis Options
C:\Users\Alpha\Desktop\Models\1ZoneEastWindow\01Defaults\1Zone_1.pic
Tone Mapping Labels Analysis Options
0 62 125 187 250 312 cd/m² absolute difference

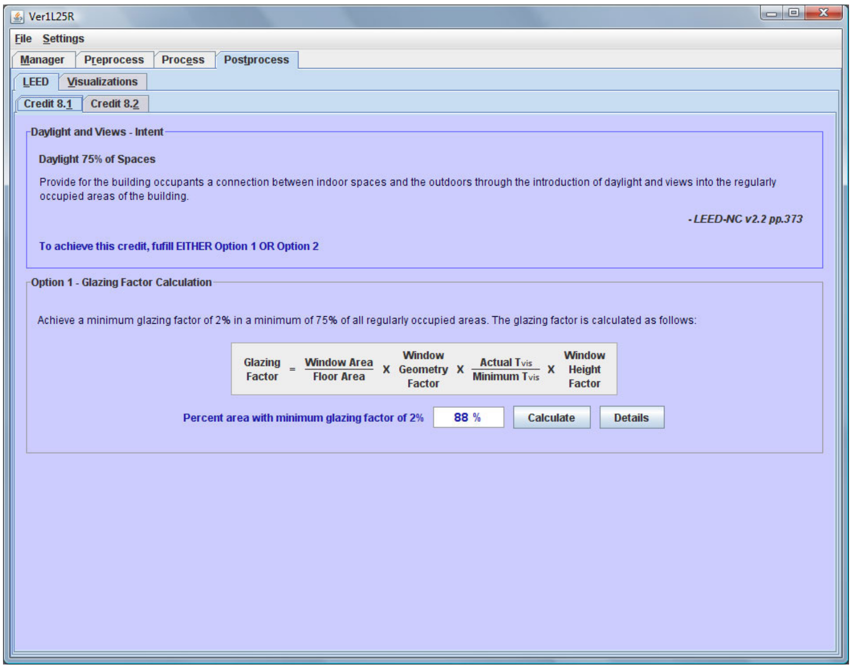
Figure 6-21 RGBE Comparator for representing difference between 2 sets of simulation results

6.2.2.10. LEED Calculators Module

The LEED Calculators Module implements the algorithms presented earlier (Chapter 5) to facilitate the automatic calculation of the two LEED credits. Since the DOM is well-formed and complete at all times (6.2.2.7), the calculators can be used once a BIM has been imported into the tool. In keeping with the GUI design of reinforcing the simulation work process (6.2.2.3), the two calculators are accessible via two nested tabs in the GUI, arranged hierarchically below the LEED sub-tab, underneath the post-processing tab (Figure 6-22 and Figure 6-23).

In both calculator-tabs, the GUIs are kept to a minimal in line with the overall design of minimizing the need for user input. In each tab, there are only two user buttons, with basic explanation of the LEED credit being calculated, and a single text field displaying the calculation result. The two user buttons correspond to the need for the tool to relevant to both preliminary as well as detailed design stages; the “Calculate” button performs the LEED credit calculation and updates the result text field, while the “Details” button brings up the calculation details (Figure 6-24 and Figure 6-25) for inspection. In the case of LEED EQ 8.2, the details GUI allows the user to edit room occupancy characteristics if the values are default values (populated by the Form Complete BIM Module if initially missing in original SOM) and require updating.

The calculation details are also formatted in LEED submittal requirements to reduce administrative time and effort. As mentioned earlier (6.2.2.3), the LEED Calculators Module is linked dynamically to the DOM Manager Module; any changes made to the DOM (by any other module) will result in an instantaneous update of the LEED evaluations.



Daylight and Views - Intent

Daylight 75% of Spaces

Provide for the building occupants a connection between indoor spaces and the outdoors through the introduction of daylight and views into the regularly occupied areas of the building.

- LEED-NC v2.2 pp.373

To achieve this credit, fufill EITHER Option 1 OR Option 2

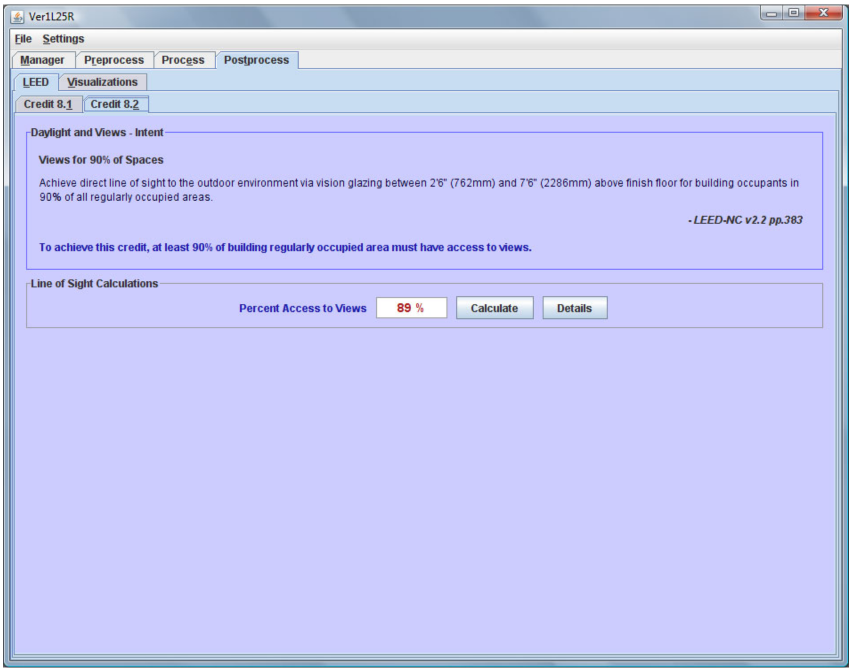
Option 1 - Glazing Factor Calculation

Achieve a minimum glazing factor of 2% in a minimum of 75% of all regularly occupied areas. The glazing factor is calculated as follows:

$$\text{Glazing Factor} = \frac{\text{Window Area}}{\text{Floor Area}} \times \text{Window Geometry Factor} \times \frac{\text{Actual } T_{vis}}{\text{Minimum } T_{vis}} \times \text{Window Height Factor}$$

Percent area with minimum glazing factor of 2%: 88 %

Figure 6-22 LEED EQ 8.1 Calculator GUI



Daylight and Views - Intent

Views for 90% of Spaces

Achieve direct line of sight to the outdoor environment via vision glazing between 2'6" (762mm) and 7'6" (2286mm) above finish floor for building occupants in 90% of all regularly occupied areas.

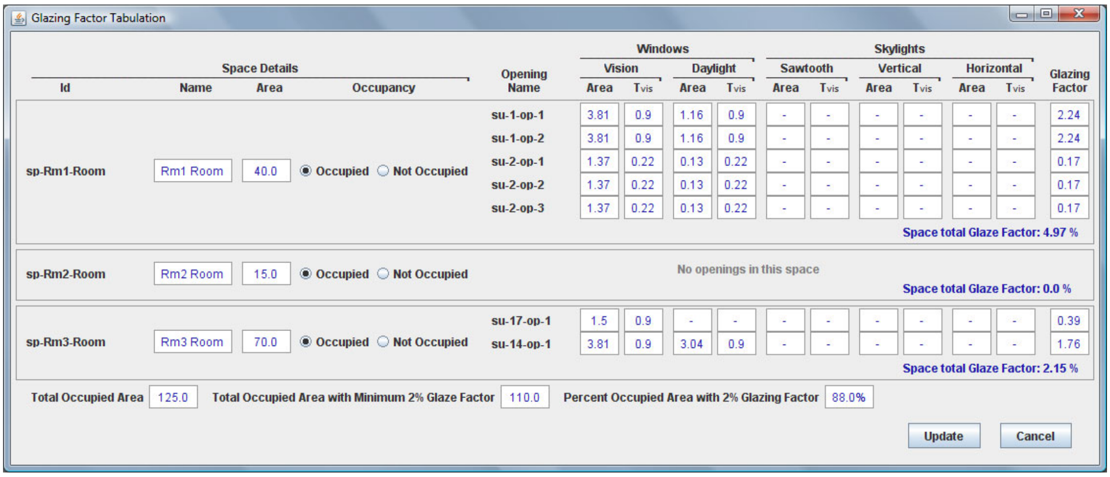
- LEED-NC v2.2 pp.383

To achieve this credit, at least 90% of building regularly occupied area must have access to views.

Line of Sight Calculations

Percent Access to Views: 89 %

Figure 6-23 LEED EQ 8.2 Calculator GUI

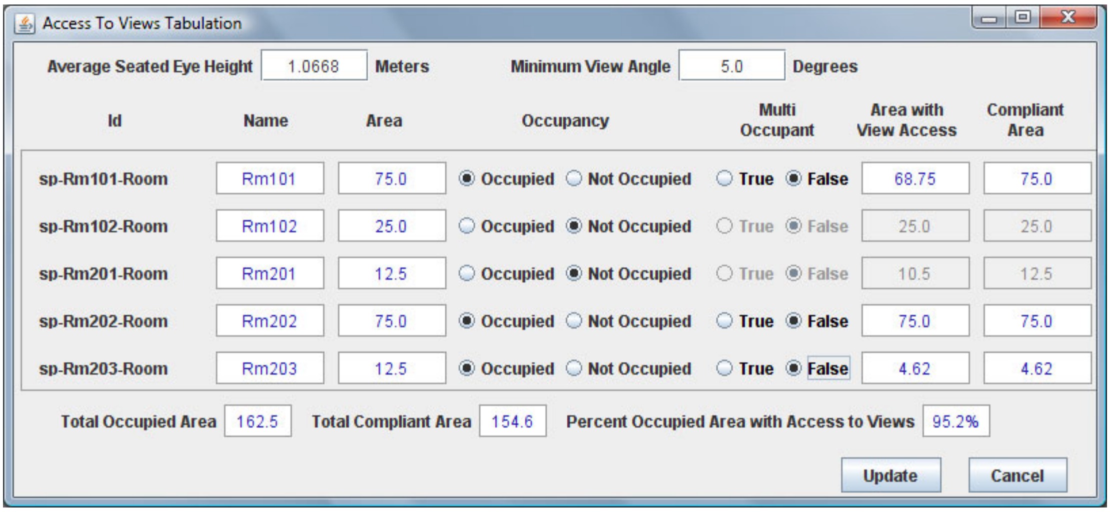


Glazing Factor Tabulation

					Windows				Skylights						
Space Details					Vision		Daylight		Sawtooth		Vertical		Horizontal		
Id	Name	Area	Occupancy	Opening Name	Area	T_{vis}	Area	T_{vis}	Area	T_{vis}	Area	T_{vis}	Area	T_{vis}	Glazing Factor
sp-Rm1-Room	Rm1 Room	40.0	● Occupied ○ Not Occupied	su-1-op-1	3.81	0.9	1.16	0.9	-	-	-	-	-	-	2.24
				su-1-op-2	3.81	0.9	1.16	0.9	-	-	-	-	-	-	2.24
				su-2-op-1	1.37	0.22	0.13	0.22	-	-	-	-	-	-	0.17
				su-2-op-2	1.37	0.22	0.13	0.22	-	-	-	-	-	-	0.17
				su-2-op-3	1.37	0.22	0.13	0.22	-	-	-	-	-	-	0.17
															Space total Glaze Factor: 4.97 %
sp-Rm2-Room	Rm2 Room	15.0	● Occupied ○ Not Occupied				No openings in this space								
															Space total Glaze Factor: 0.0 %
sp-Rm3-Room	Rm3 Room	70.0	● Occupied ○ Not Occupied	su-17-op-1	1.5	0.9	-	-	-	-	-	-	-	-	0.39
				su-14-op-1	3.81	0.9	3.04	0.9	-	-	-	-	-	-	1.76
															Space total Glaze Factor: 2.15 %

Total Occupied Area 125.0 Total Occupied Area with Minimum 2% Glaze Factor 110.0 Percent Occupied Area with 2% Glazing Factor 88.0%

Figure 6-24 LEED EQ 8.1 Calculation Details



Access To Views Tabulation

Average Seated Eye Height 1.0668 Meters Minimum View Angle 5.0 Degrees

Id	Name	Area	Occupancy	Multi Occupant	Area with View Access	Compliant Area
sp-Rm101-Room	Rm101	75.0	● Occupied ○ Not Occupied	○ True ● False	68.75	75.0
sp-Rm102-Room	Rm102	25.0	○ Occupied ● Not Occupied	○ True ● False	25.0	25.0
sp-Rm201-Room	Rm201	12.5	○ Occupied ● Not Occupied	○ True ● False	10.5	12.5
sp-Rm202-Room	Rm202	75.0	● Occupied ○ Not Occupied	○ True ● False	75.0	75.0
sp-Rm203-Room	Rm203	12.5	● Occupied ○ Not Occupied	○ True ● False	4.62	4.62

Total Occupied Area 162.5 Total Compliant Area 154.6 Percent Occupied Area with Access to Views 95.2%

Figure 6-25 Tabulation of LEED Credit EQ 8.2

6.2.2.11. RADIANCE Input Files Generator Module

The RADIANCE Input Files Generator Module allows the user to obtain a set of simulation input files for RADIANCE from the imported BIM without any intervention. This feature facilitates the checking of simulation results from the implemented new simulation technique, as well as provides the functionality to generate photo-realistic images with minimal user effort. Since the DOM is well-formed and complete at all times, all necessary information is present and the module only has to translate the gbXML formatted DOM into the RADIANCE format. A set of batch files is also generated so that the user can initiate RADIANCE simulation without additional work.

Besides a syntactical translation, the module automates the ontological mapping between the gbXML-based DOM and RADIANCE schema. While gbXML follows real-world descriptors of relational elements (a window is located on a wall, thus the wall is a parent surface of a child window surface; both surfaces have 4 vertices if rectangular and intersects), RADIANCE descriptions strictly define geometric scenes (there is no hierarchical difference between wall and window surfaces; the rectangular wall has 4 additional vertices defining a void where the window is, the two surfaces do not intersect). RADIANCE descriptions discriminates surfaces based upon the impact on the Monte-Carlo sampling technique (4.1.1); surfaces with anticipated high luminance and frequency distribution have a different categorical definition.

The module pre-processes all elements within the DOM, identifying surfaces that are categorically different in RADIANCE. Parent surfaces containing child surfaces are modified by inserting vertices and defining voids so that they no longer overlap child surfaces. Once the information is ontologically consistent with the RADIANCE schema, the module then translates the model syntactically and generates all necessary auxiliary files necessary for RADIANCE simulation.

```
GenerateRADIANCEFiles() {
    sortSurfaces()              identifies surfaces that need to be defined differently in RADIANCE
                                (surfaces that should be defined as primary and secondary sources)
    processSurfaces()           triangulates affected parent surfaces and inserts vertices to define voids
    processSources()            generates primary and secondary light sources in RADIANCE syntax
    generateRadianceScene()     generates scene geometry in RADIANCE syntax
    generateAux()               generates material, sky, and camera definition files,  batch files
}
```

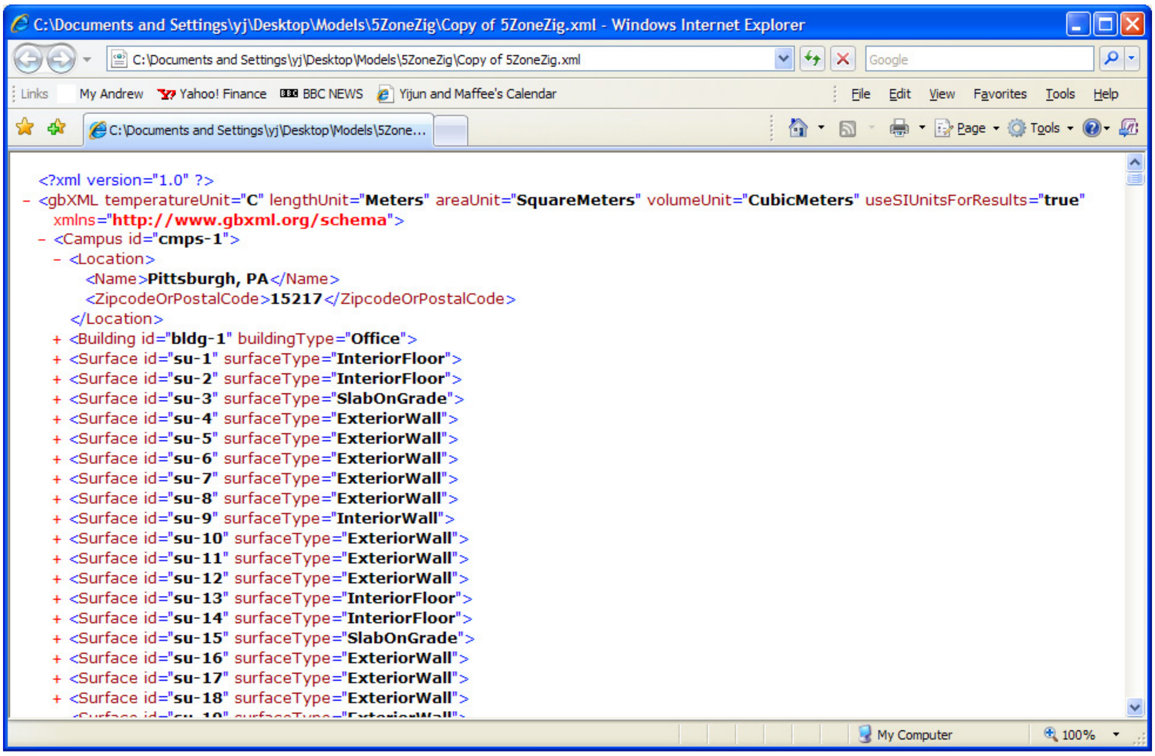
Figure 6-26 Pseudocode for generating RADIANCE input files

6.2.3. Scalability across Design Stages - Maintaining a completed DOM

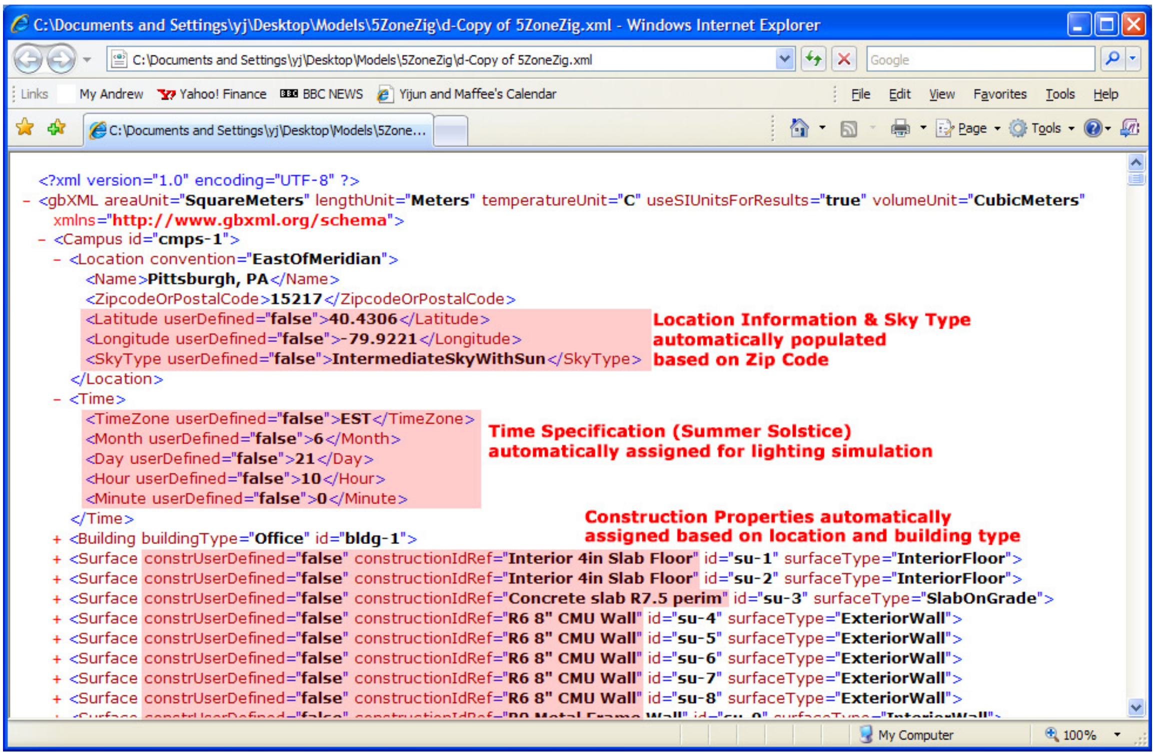
The use of using appropriate placeholders (6.2.2.6) ensures a complete and well-formed model at all times to overcome the problem of incomplete or missing information due to the LOD of design. This allows the use of a consistent benchmarks and metrics throughout the design process. By completing the SOM with information from the implemented external datasets, consistency between the various DOMs is ensured. This enhances concurrency since dependency between tasks is reduced, and the potential downstream problem of impossible specifications and products is avoided.

Given the fact that the building industry typically employs a relatively limited variety of standard practices, the preparation of this shared data set is relatively straightforward. A heuristic rule-based algorithm is used to populate the SOM by querying the shared datasets according to available information and the context of such information. In cases where the required information is absent from the dataset, a nearest-neighbor search allows the selection of appropriate values (Figure 6-27). The entire process is automatic and instantaneous; there is no need for user intervention to review the SOM or search for appropriate and consistent assumptions, the time and effort to prepare a well-formed lighting model is greatly reduced.

The problem of LOD, where information in the DOM is missing due to the stage of design, is essentially reframed as that of level of confidence (LOC), where the BIM is used keeping in mind that certain assumptions have been made. Scalability is achieved since a consistent methodology is used regardless of the level of information availability and precision. The use of LOC may also be more consistent with professional practice considerations such as due diligence and consistent with the progressive nature of design.


C:\Documents and Settings\yj\Desktop\Models\5ZoneZig\Copy of 5ZoneZig.xml - Windows Internet Explorer
Links
My Andrew
Yahoo! Finance
BBC NEWS
Yijun and Maffee's Calendar
File
Edit
View
Favorites
Tools
Help
Page
Tools
My Computer
100%

```
  <?xml version="1.0" ?>
- <gbXML temperatureUnit="C" lengthUnit="Meters" areaUnit="SquareMeters" volumeUnit="CubicMeters" useSIUnitsForResults="true"
    xmlns="http://www.gbxml.org/schema">
  - <Campus id="cmps-1">
    - <Location>
        <Name>Pittsburgh, PA</Name>
        <ZipcodeOrPostalCode>15217</ZipcodeOrPostalCode>
      </Location>
    + <Building id="bldg-1" buildingType="Office">
    + <Surface id="su-1" surfaceType="InteriorFloor">
    + <Surface id="su-2" surfaceType="InteriorFloor">
    + <Surface id="su-3" surfaceType="SlabOnGrade">
    + <Surface id="su-4" surfaceType="ExteriorWall">
    + <Surface id="su-5" surfaceType="ExteriorWall">
    + <Surface id="su-6" surfaceType="ExteriorWall">
    + <Surface id="su-7" surfaceType="ExteriorWall">
    + <Surface id="su-8" surfaceType="ExteriorWall">
    + <Surface id="su-9" surfaceType="InteriorWall">
    + <Surface id="su-10" surfaceType="ExteriorWall">
    + <Surface id="su-11" surfaceType="ExteriorWall">
    + <Surface id="su-12" surfaceType="ExteriorWall">
    + <Surface id="su-13" surfaceType="InteriorFloor">
    + <Surface id="su-14" surfaceType="InteriorFloor">
    + <Surface id="su-15" surfaceType="SlabOnGrade">
    + <Surface id="su-16" surfaceType="ExteriorWall">
    + <Surface id="su-17" surfaceType="ExteriorWall">
    + <Surface id="su-18" surfaceType="ExteriorWall">
    + <Surface id="su-19" surfaceType="ExteriorWall">
```


C:\Documents and Settings\yj\Desktop\Models\5ZoneZig\d-Copy of 5ZoneZig.xml - Windows Internet Explorer
Links
My Andrew
Yahoo! Finance
BBC NEWS
Yijun and Maffee's Calendar
File
Edit
View
Favorites
Tools
Help
Page
Tools
My Computer
100%

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <gbXML areaUnit="SquareMeters" lengthUnit="Meters" temperatureUnit="C" useSIUnitsForResults="true" volumeUnit="CubicMeters"
    xmlns="http://www.gbxml.org/schema">
  - <Campus id="cmps-1">
    - <Location convention="EastOfMeridian">
        <Name>Pittsburgh, PA</Name>
        <ZipcodeOrPostalCode>15217</ZipcodeOrPostalCode>
        <Latitude userDefined="false">40.4306</Latitude>
        <Longitude userDefined="false">-79.9221</Longitude>
        <SkyType userDefined="false">IntermediateSkyWithSun</SkyType>
      </Location>
    - <Time>
        <TimeZone userDefined="false">EST</TimeZone>
        <Month userDefined="false">6</Month>
        <Day userDefined="false">21</Day>
        <Hour userDefined="false">10</Hour>
        <Minute userDefined="false">0</Minute>
      </Time>
    + <Building buildingType="Office" id="bldg-1">
    + <Surface constrUserDefined="false" constructionIdRef="Interior 4in Slab Floor" id="su-1" surfaceType="InteriorFloor">
    + <Surface constrUserDefined="false" constructionIdRef="Interior 4in Slab Floor" id="su-2" surfaceType="InteriorFloor">
    + <Surface constrUserDefined="false" constructionIdRef="Concrete slab R7.5 perim" id="su-3" surfaceType="SlabOnGrade">
    + <Surface constrUserDefined="false" constructionIdRef="R6 8" CMU Wall" id="su-4" surfaceType="ExteriorWall">
    + <Surface constrUserDefined="false" constructionIdRef="R6 8" CMU Wall" id="su-5" surfaceType="ExteriorWall">
    + <Surface constrUserDefined="false" constructionIdRef="R6 8" CMU Wall" id="su-6" surfaceType="ExteriorWall">
    + <Surface constrUserDefined="false" constructionIdRef="R6 8" CMU Wall" id="su-7" surfaceType="ExteriorWall">
    + <Surface constrUserDefined="false" constructionIdRef="R6 8" CMU Wall" id="su-8" surfaceType="ExteriorWall">
    + <Surface constrUserDefined="false" constructionIdRef="R0 Metal Frame Wall" id="su-9" surfaceType="InteriorWall">
```

Location Information & Sky Type automatically populated based on Zip Code

Time Specification (Summer Solstice) automatically assigned for lighting simulation

Construction Properties automatically assigned based on location and building type

Figure 6-27 Heuristic rule-based automatic population of DOM
Original information (top), completed DOM (bottom)

6.3. Use of New Tool within Adaptive-Iterative Design Process

To evaluate the use of the new tool within adaptive-iterative design processes, two separate scenarios representative of the lighting design related tasks are presented:

1. First-time processing of significant version/change (iterative)
2. Modification of model property for optimization (adaptive)

Adaptive and iterative processes are not mutually exclusive and design efforts encompass traits from both, defying ease of simplistic categorization. However, the defining trait of the iterative process is a cyclic process of synthesizing, testing, evaluating, and improving a design solution, which involves distinct solutions. The adaptive process focuses on the continual evolution of a design solution to improve its performance. Within this context, the two processes are used to characterize the two design conditions where the primary tasks are to either consider a new design version, or to compare an incremented design change.

The following evaluation of user time-effort savings includes features provided in the new tool (automatic LEED calculators, results analysis features, and automated preparation of simulation input files, but excluding simulation time) to alleviate lighting design efforts, since user effort is associated only with manual tasks and does not include CPU-time.

6.3.1. Iterative Process

In the iterative-design scenario, a design solution that is new or significantly different from previous versions is considered in the lighting domain. In such cases, the primary task is to prepare a working lighting domain model followed by performance evaluation and analysis. In contemporary practice, the preparation of the lighting domain model alone takes hours or even days. The new tool essentially eliminates this task.

In contemporary practice the design model would have to be remodeled into a lighting model, with significant effort to gather location, construction, and material information from disparate sources. The entire process would also involve different tools for simulation, data analysis, and performance metric calculations.

In the case of the new tool the design model, in the form of a BIM, can be directly imported and ready for simulation without any need for user-intervention. All the processes, including lighting simulation, data post-processing features, and performance metric (LEED Credits EQ 8.1 & EQ 8.2) calculations, are all available within the tool and similarly do not require any user intervention, though all parameters can be inspected and edited if desired.

By reducing the time and effort required to evaluate a design solution, the new tool supports the iterative-design process by allowing more iterations within the same time, effort, and cost constraints.

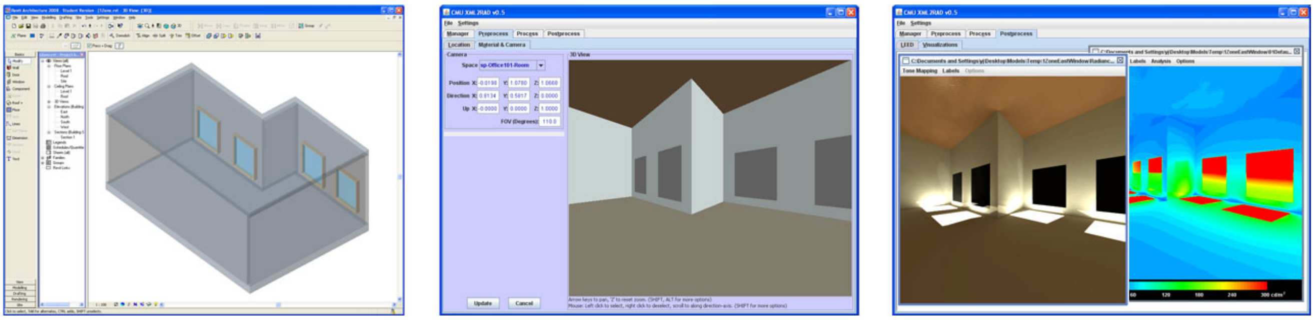

CMU Xml 2RAD v0.5
File Settings
Manager Preprocess Process Postprocess
Location Material & Camera
3D View
Update
Cancel
LEED Visualizations
Tone Mapping Labels Options
Labels Analysis Options

Figure 6-28 First Scenario.
Existing building model from external CAD software (left). Automatic processing and population of missing values instantaneously and without user intervention (middle). Simulation Results and false-color analysis (right).

6.3.2. *Adaptive Process*

In the adaptive-design scenario, slight modifications to an existing design are made to form a better understanding of the existing design solution, test ideas, and identify possible optimization. In this case, updates in the performance metrics should be fast while the accuracy of the updated performance metrics does not always have to be absolute since they are used mainly comparatively with the base results.

The new tool supports adaptive design by providing features to quickly evaluate the performance changes due to design modifications, as well as features to compare the difference in performance. In the case of the former, the LEED credits calculators are dynamically updated as changes are made to the building model (Figure 6-29). The user can try different parameter variations intuitively before committing the desired state back to the DOM. Likewise, the direct sampling lighting simulation technique (Chapter 4.4) also provides physically-accurate radiance results and ensuing comparisons quickly. To facilitate ease of comparisons, a results comparator (Figure 6-21) is also provided.

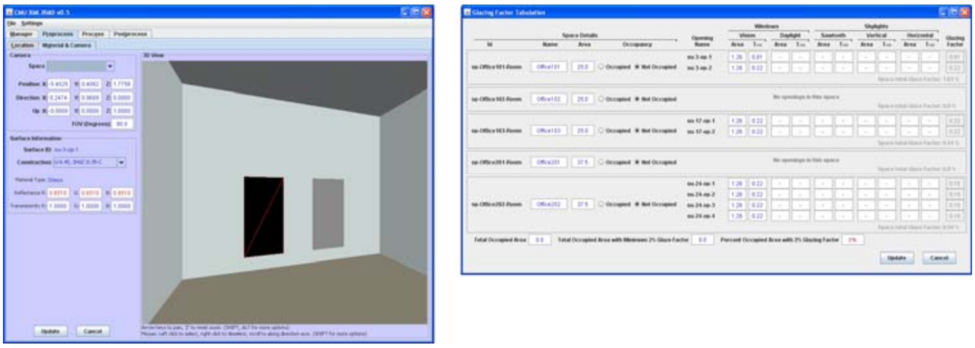
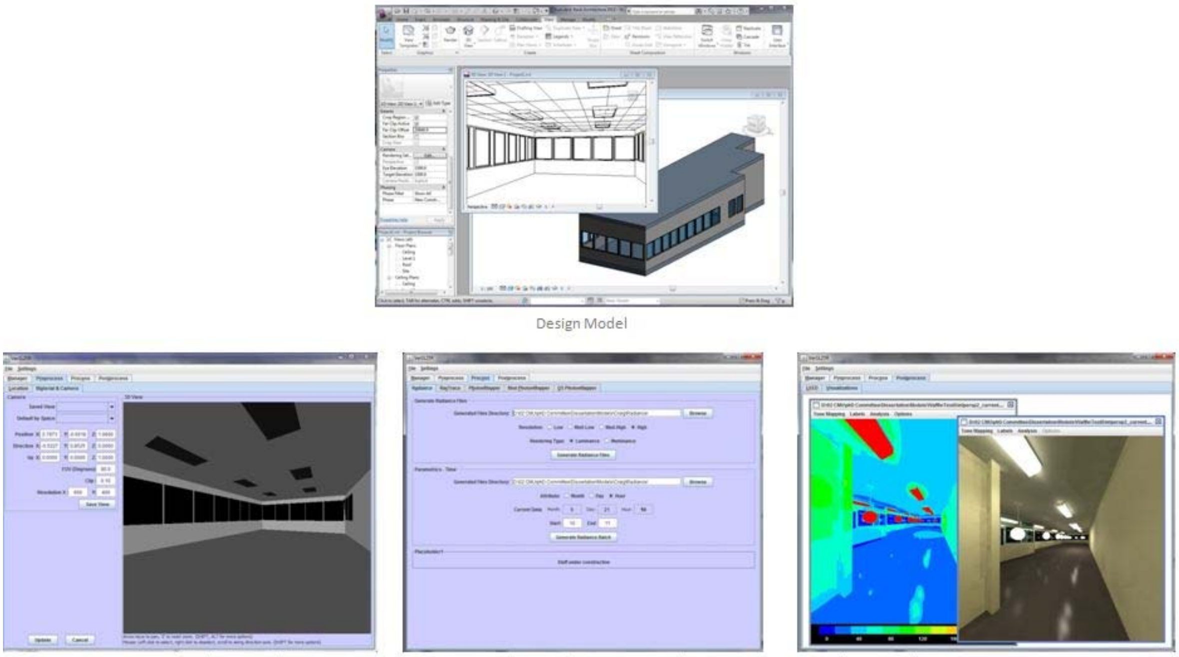


Figure 6-29 Second Scenario.
Inspection and editing of window construction properties (left) and updated LEED credit tabulation (right).

6.4. Comparison of New Tool to Case Studies



Design Model

Automatic Preparation of Lighting Simulation Model

Automatic Radiance File Generator and Parametric Batch Files

Simulation Results Post Processing

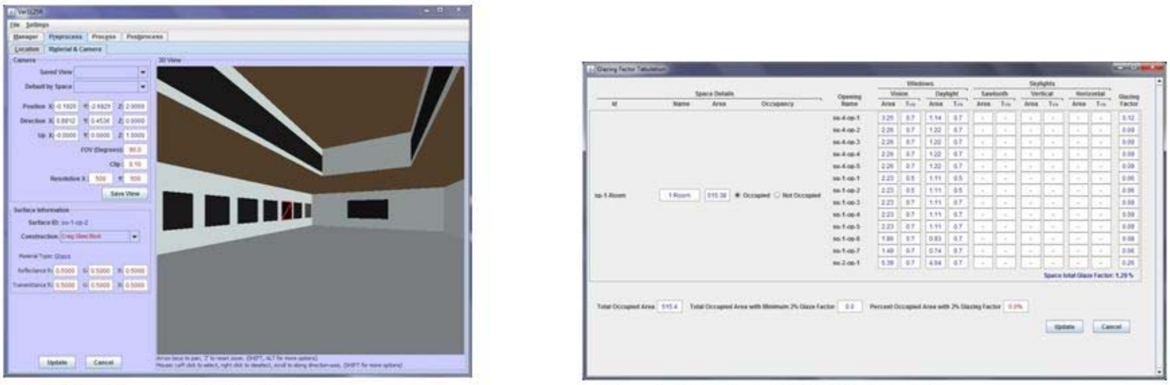
Figure 6-30 Automatic RADIANCE simulation input files preparation by new tool (Case Study 1)

		Manual Process	**New Lighting Tool**
		- *All information in CAD drawings and spreadsheets* - *Expert User*	- *Well formed and complete BIM* - *Novice User*
Simulation Input Files	Geometry	7 Hours	< 1 Sec
	Material Information	1 Hour	
	Lamp Properties	0.5 Hour	
	Sky Definition	0.5 Hour	
Simulation Parameters	2 Camera Views	0.1 Hour	< 0.1 Hour
	Batch Files	0.5 Hour	
Simulation Run	RADIANCE Computation time	24 Hours (1 hour each time-step)	
Post processing	Generate false-color images from results	3 Hours	
Iterative Run	Update Geometry and Lamps in Input Files Generate new batch file	2 Hours	< 1 Sec
Simulation Run	RADIANCE Computation time	24 Hours (1 hour each time-step)	
Post processing	Generate false-color images from results	3 Hours	
Iterative Run	…		

Table 6-10 Comparison of time to obtain simulation input files (Case Study 1)

Recalling the simulation workflow in Case Study 1 (Chapter 3.1), the new tool essentially eliminates the user effort in preparing the simulation input files. Given a well-formed and complete BIM prepared using Revit (Figure 6-30, top), the new tool automatically populates any missing information, and generates RADIANCE simulation input files at the click of a single button. This process takes less than 1 second, and does not require any expertise with simulation syntax or processes (Table 6-10). The new tool also automatically inserts a camera-view in each space, which can be user inspected and edited (Figure 6-30, bottom left) via the GUI. Likewise, the new tool provides a GUI (Figure 6-30, bottom middle) to automatically generate the necessary sets of simulation input files and a batch file for parametric time-step simulation. The inspection and definition of these simulation parameters (cameras and batch files) take less than 0.1 hour.

Significant time savings can also be similarly achieved in subsequent iterative runs, assuming the same availability of well-formed and complete BIM. Ideally, the new tool should incorporate geometric editing so that design changes motivated by simulation results can immediately be effected. However, the new tool at this point only supports material and lamp properties editing; geometry changes have to be made in the design modeling tool. Even so, since the new tool is interoperable (via the gbXML schema) with the design modeling tool (Revit), the new design can be imported into the new tool with relative ease.

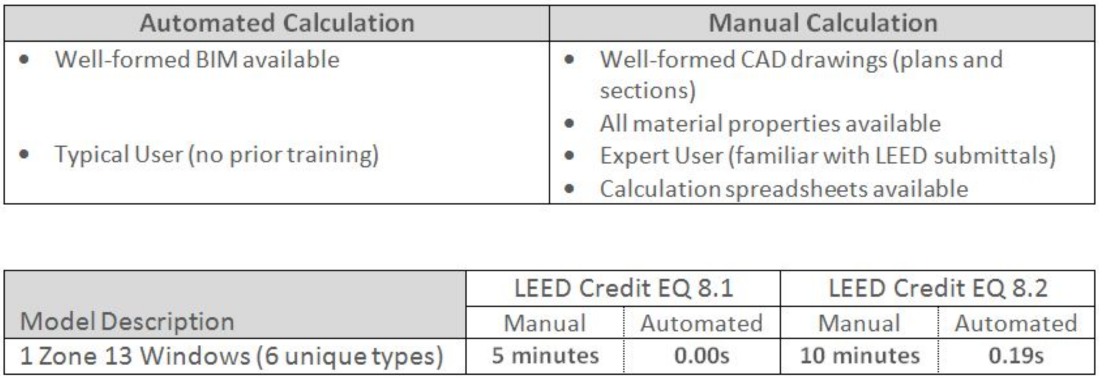


Design Model imported into New Tool

Automatic Glazing factor (LEED EQ8.1) Calculation

Figure 6-31 Automatic LEED Credit calculation by new tool (Case Study 2)

Case Study 2 involves 2 tasks, the calculation of LEED Credits EQ 8.1 and 8.2, and day lighting simulation. Given a well-formed and complete BIM prepared using Revit, new tool automatically calculates both LEED credits almost instantaneously (Table 6-11). Similar to Case Study 1, the new tool essentially eliminates the user effort in preparing the simulation input files, achieving significant time and user effort reductions even when compared to using middleware such as Ecotect (Table 6-12).



Automated Calculation	Manual Calculation
• Well-formed BIM available • Typical User (no prior training)	• Well-formed CAD drawings (plans and sections) • All material properties available • Expert User (familiar with LEED submittals) • Calculation spreadsheets available

Model Description	LEED Credit EQ 8.1		LEED Credit EQ 8.2	
	Manual	Automated	Manual	Automated
1 Zone 13 Windows (6 unique types)	5 minutes	0.00s	10 minutes	0.19s

Table 6-11 Comparison of time to calculate LEED Credits (Case Study 2)

		Manual Process	Ecotect	New Lighting Tool
		• *All information in CAD drawings and spreadsheets* • *Expert User*	• *Successful CAD drawings import* • *Expert User*	• *Well formed and complete BIM* • *Novice User*
Simulation Input Files Preparation	Geometry	4 Hours	10 Minutes	< 1 Sec
	Material Properties	0.5 Hour		
	Sky & Camera Definition	0.1 Hour		
	Batch Files	0.5 Hour		
Simulation Run	RADIANCE Computation time	4.5 Hours (15 Minutes each time-step)		
Post processing	Generate false-color images	3 Hours		
Iterative Run	Update Geometry Generate new batch files	1 Hour	10 Minutes	< 1 Sec
Simulation Run	RADIANCE Computation time	4.5 Hours (15 Minutes each time-step)		
Post processing	Generate false-color images	3 Hours		
Iterative Run	…			

Table 6-12 Comparison of time to obtain simulation input files (Case Study 2)

7. Conclusions

The new tool is checked against the research objectives and areas of future work discussed.

7.1. Conclusions

As mentioned in the introduction, this research is interested in reducing the time and effort in conducting lighting simulation, and brings together concepts from various domains to prototype a new tool that automates much of the simulation workflow. There are two immediate limitations to the work in this research being limited to tool-building following and building upon theories in existing literature.

Given that the ultimate goal of the tool is to be used in real-world building design scenarios, empirical validation is an equally important counterpart to the theory-based work presented here. While the work presented hitherto represents necessary ground work and integration to start building an interoperable, highly automated, first principle-based lighting simulation tool, the same development needs to be informed and supported by actual user testing and feedback. While the new tool has met research objectives, it makes no claim on effectiveness or usefulness in practical usage; a parallel set of empirical research effort, including user testing, needs to be conducted.

Likewise, the second limitation of this research pertains to the lack of empirical validation in supporting the proposed simulation techniques. While the proposed Photon Mapping technique is adapted from well-established research, extensive and generalized validation is required to conclusively determine if the suggestion to reduce approximations or importance sample will be useful, that the observed differences are trending towards desired accuracy. Given the complexity of high performance designs, analytical validation would be unfeasible; empirical validation using physical models would be the most effective validation method.

Topics in this necessary body of empirical research are discussed in the following section (7.3).

Nonetheless, this research posits that a new interoperable design support tool that provides automatic simulation, post-processing, and benchmark calculations, achieved by integrating and adapting such existing technologies, will be fast and easy to use. The research objectives (Chapter 1.4) were:

1. Reduce the time and effort to set up and conduct lighting simulation
2. Complement integrated design processes
3. Use consistent performance metrics and technical approaches
4. Provide operative information with minimum user effort.
5. Implement a first principle-based rendering engine

The new tool achieves near-effortless preparation for lighting simulation by automating the entire process, utilizing interoperability with Building Information Models, shared project-wide databases, and maintaining a well-formed Domain Object Model (DOM) at all times. The time to prepare for lighting simulation has been reduced to mere seconds. Usability is addressed by minimizing the number of actions users go through when using each feature, although all parameters can be inspected and edited when desired. Color coding is also used to cognitively communicate the state of the model.

The new tool supports models of varying completeness by automatically populating missing information in the DOM using real world-based project-wide datasets. The new tool achieves interoperability with other tools by supporting data exchange with an external project-wide Shared Object Model (SOM). This is a step towards facilitating concurrent, multiple domain, activities necessary in integrated design, since this modular approach allows other domain activities to proceed independently. By completing the Shared Object Model (SOM) with information from the project-wide datasets, consistency between the various DOMs is also ensured. This enhances concurrency since dependency between tasks is reduced, and avoids problems associated with conventional work flows.

By maintaining a complete DOM at all times, the new tool allows consistent benchmarks and metrics to be used throughout all design stages. Scalability, the applicability of the new tool across all stages of design, is achieved since consistent calculation and simulation methods are used regardless of the level of information availability and precision.

To meet the objective of operative information, automatic calculators for two LEED lighting performance benchmarks commonly used in the United States are implemented. The calculators demonstrate the dynamic provision of consistent operative information for design decisions throughout the design process, regardless of the LOD in the design information. By presenting the results in submittal formats, documentation time is also reduced. Simulation results analysis and visualization features useful to typical lighting analysis and design tasks including normalized luminance ratios and data comparators are also implemented.

To meet the last objective, the Photon Mapping technique (Jensen, 2001) is implemented for its ability to solve the global illumination rendering equation, and the advantage over Monte Carlo (backwards) ray-tracing in dealing with highly reflected irradiance conditions typical in high performance design. The classic (Jensen) technique is recognized as formulated for efficient computation for photo-realistic images, and several approximations are suggested to be removed for the technique to be better suited to the architecture design context. Similarly, the context of use suggests the possibility of using importance sampling. While the modified technique still requires validation, it managed to render a demonstrative scene where RADIANCE was unable to do so (Figure 4-10), demonstrating the potential to consider high performance building design features. At the same time, the progressive nature of the implemented technique also allows the simulation engine to be scalable (in terms of computational time) according to the desired LOD. Finally, a fast visualizations mode suitable for comparative design scenarios is explored.

7.2. Contributions

		Design Support Tools		Simulation Engines	
	Features	New Lighting Simulation Tool	Ecotect	Radiance	Photon Mapping
Design Support Tool Features	Import/Export BIM	√	Limited		
	Project-wide Data Libraries	√	-		
	Automatic Completion of Partial Models (LOD)	√	Limited		
	Modeling Capabilities	-	√		
	Feedback on accuracy	√	-		
	Automatic Code Checking	LEED	UK Part-L		
	Dynamic Feedback	√	-		
	Link to physically-accurate simulation engine	√ (Internal)	√ (Radiance)		
	Simulation results visualization	√	√		
Simulation Engine Features	Physically accurate	√		√	√
	Progressive	√		-	√
	Supports Typical High Performance Building Designs	√		-	Not Optimized
	Supports Fast Visualizations for Comparative Studies	√		-	Not Optimized

Table 7-1 Summary of features, comparison with existing design support and simulation tools

The new tool demonstrates the feasibility of making lighting simulation more accessible and useful to the architect. Whereas disparate tools have to be learned and used for each part of the simulation workflow previously, the new tool brings the features together in a single package, designed with an emphasis on the operative information required for design decisions. Following the philosophy of Ecotect to be "*written and developed by architects with its application in architecture and the design process firmly in mind*", the new tool purports to contribute to this effort. By making it easy to perform lighting simulations as part of the design process, it is the agenda of this research to facilitate the investigation, development, and synthesis, of high performance building designs.

Any environmental improvement desired of buildings is only possible if that quality can be consistently measured. In the development of the LEED calculators, ambiguities in the benchmarks, areas and methods for improvements, have been highlighted.

Despite the age of the Photon Mapping technique and its popularity and prevalence in other disciplines, there is very little effort in building sciences to capitalize on it. Only one research case (Schregle, 2005) was found, and only the conventional technique (Jensen, 2001) was implemented. The efforts in this research contribute towards developing a model suitable for use in the building design context.

7.3. Future Work

At the time of writing, the author has embarked on research to investigate the following topics. The motivation, like this research, remains the development of easy to use lighting design support tools. While some of the topics are computational in nature, the goal puts the work in context.

7.3.1. Validation of Modified Photon Mapping Technique

As mentioned, there is a need for empirical validation and user testing to ensure suitability of use for evaluating high performance buildings. While this research presented a general case of highly-reflected scenes commonplace in high performance buildings (Figure 4-2, Figure 4-11 and Figure 4-23), a larger set of test-models would ensure all pertinent conditions in high performance building designs being comprehensively captured, and the implemented engine sufficiently robust and dependable for use. Empirical testing and validation are also necessary to provide the level of confidence required for actual real-world application. Validation methodologies, both analytical (Witte, Henninger, & Glazer, 2001; Reinhart & Herkel, 2000) as well as empirical (Maamari, 2006) are well established and should used.

The accuracy of a simulation tool also depends upon how it is applied and used, besides the correctness of the fundamental formulations. The simulation parameter settings and approach of preparing input definitions have been shown (Lam, Huang, & Zhai, 2004) to cause significant variations in simulation output. Unfortunately, much tacit expert knowledge is required to ensure correctness of input. As part of the validation work, sensitivity analyzes can also be used to characterize the impact of various parameters within the architecture design simulation context.

7.3.2. Sky Luminance Mapping

The current implementation of the simulation engine does not yet include a sky-source. Empirical sky models are well established (Commission Internationale de l'Eclairage, 2003) and use two functions to describe the luminance gradation of a sky as across a hemispherical projection. Unfortunately, a simple implementation of the sky model (hemispherical light source over the building geometry) would result in an unfeasibly large numbers of photons. It might be possible to use projections from the building exterior openings and obtain a distribution function for each of the openings, treating them as the light sources, or to investigate the feasibility of approximation techniques such as using a *light tree* (Walter, Fernandez, Abree, Bala, Donikian, & Greenberg, 2005) within the architecture lighting design context.

7.3.3. User Testing

While the automation of previously manual tasks obviously reduces time and effort significantly, the user-friendliness of the new tool has not been empirically assessed. User tests in actual building design contexts should be conducted to validate and improve the design of the UI, as well as the implicit work-process governed by the framework design of the new tool. Similarly, the automatic LEED credits calculators are only a first demonstration of providing operative information in a timely fashion; user feedback has yet to be collected. Other issues, notable user-centric issues such as ease-of-use, interpretation of results accuracy, application of results in decisions, are not yet covered by this research.

.

References

Acdx. (2009, March 15). *The CIE Standard Observer Color Matching Functions [Online image].* Retrieved April 15, 2009, from http://en.wikipedia.org/wiki/File:CIE_1931_XYZ_Color_Matching_Functions.svg

Augenbroe, G. (2001). Building Simulation Trends Going Into The New Millennium. *Proceedings of Building Simulation '01* (pp. 15-27). Rio de Janeiro: IBPSA.

Augenbroe, G., & Winkelmann, F. (1991). Integration of simulations into the design process. *Proccedings of Building Simulation '91* (pp. 367-734). Nice: IBPSA.

Autodesk. (2009). Ecotect 2009 [Software].

Autodesk. (2009). Green Building Studio [Software].

Autodesk. (1999). Lightscape Version 3.2 [Software].

Autodesk. (2009). Revit Architecture [Software].

Bentley, J. L. (1975). Multidimensional Binary Search Trees used for Associative Searching. *Communications of the ACM, Vol. 18, No. 9* , 509-517.

Bryan, H., & Mohammed Autif, S. (2002). Lighting/Daylighting Analysis: A Comparison. *Proceedings of the Solar Conference.* Reno, Nevada.

Center for Total Building Performance. (2003). *Regulations and Guidelines for Total Building Performance*. Retrieved 8 31, 2009, from Center for Total Building Performance: http://www.bdg.nus.edu.sg/BuildingEnergy/about_eric/regulations.html

Christensen, P. H. (1999). Faster Photon Map Global Illumination. *Journal of Graphics Tools, 4(3)* , 1-10.

City of New York Department of Design and Construction. (April, 1999). *High Performance Building Guidelines.*

Commission Internationale de l'Eclairage. (2003). *Spatial Distribution of Daylight: CIE Standard General Sky. CIE S 011/E:2003.* Vienna: Commission Internationale de l'Eclairage.

Commission Internationale de l'Eclairage. (1995). *Discomfort Glare in Interior Lighting. CIE 117-1995.* Wien: CIE.

Commission Internationale de l'Eclairage. (2001). *Lighting of Indoor Work Places. CIE Standard S008/E.* Wien: CIE.

Commission Internationale de l'Eclairage. (1931). The Standard Observer. *Commission Internationale de l'Eclairage Proceedings, 1931.* Cambridge: Cambridge University Press.

Deru, M., & Torcellini, P. (2004). *Improving Sustainability of Buildings Through a Performance-based Design Approach.* Colorado: National Renewable Energy Laboratory.

Design Computing, Georgia Institute of Technology. (2009, August). *Exchange Capability.* Retrieved November 23, 2009, from BIM Resources @ Georgia Tech: http://bim.arch.gatech.edu/content_view.asp?id=406

Dong, B., Lam, K. P., Huang, Y. C., & Dobbs, G. (2007). A Comparative Study of the IFC and gbXML Informational Infrastructures for Data Exchange in Computational Design Support Environments. *Proceedings of Building Simulation 2007* (pp. 1530-1537). Beijing: IBPSA.

Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2008). *BIM Handbook: A Guide to Building Information Modeling.* New Jersey: Wiley.

Energy Independence and Security Act of 2007, Public Law 110-140.

Estes, J., Schreppler, S., & Newsom, T. (2004). Daylighting prediction software: comparative analysis and application. *Proceedings of Fourteenth Symposium on Improving Building Systems in Hot and Humid Climates*, (pp. 259-267). Texas.

FIATECH. (2007). *aecXML.* Retrieved June 27, 2007, from http://www.fiatech.org/projects/idim/aecxml.htm

Fisk, W. J. (2000). Health and Productivity Gains from Better Indoor Environments and Their Relationship with Building Energy Efficiency. *Annual Review of Energy and the Environment, Vol. 25* , 537-566.

Gail, L., Todd, J. A., & Hayter, S. J. (2003). *A Handbook for Planning and Conducting Charrettes for High-Performance Projects.* . Colorado: National Renewable Energy Laboratory.

GeoPraxis Inc. (2003, August 19). *gbXML version 0.34.* Retrieved December 31, 2008, from Green Building XML Schema: http://www.gbxml.org

Hachisuka, T., Ogaki, S., & Jensen, H. W. (2008). Progressive Photon Mapping. *ACM SIGGRAPH Asia 2008.* New York: ACM.

Haines, E. (1989). Essential Ray Tracing Algorithms. In A. Glassner, *An Introduction to Ray Tracing* (pp. 33-77). London: Academic Press.

Heckbert, P. S. (1990). Adaptive Radiosity Textures for Bidirectional Ray Tracing. *Computer Graphics (Proceedings of SIGGRAPH '90), Vol. 24, No. 4* , 145-154.

Hedge, A., Sims, W. K., & Becker, F. D. (1995). Effects of lensed-indirect and parabolic lighting on the satisfaction, visual health, and productivity of office workers. *Ergonomics, Vol. 32, No. 2* , 260-280.

Heshong, L., Wright, R. L., Okura, S., Klein, P. D., Simner, M., Berman, S., et al. (2002). Daylighting impacts on human performance in school. *Journal of the Illuminating Engineering Society Vol. 32, No. 2* , 101-114.

High Performance Building Congressional Caucus Coalition. (2009, April 16). *Producing High Performance Federal Buildings.* Retrieved May 4, 2009, from High Performance Building Congressional Caucus Coalition: http://www.hpbccc.org/

Illuminating Engineering Society of North America. (2000). *Lighting Handbook. Reference & Applications, 9th Edition.* New York: IESNA.

International Alliance for Interoperability. (2008a, September 12). *Frequently Asked Questions.* Retrieved November 23, 2009, from BuildingSmart: http://www.iai-tech.org/services/faq/fag-general-ifc-spec

International Alliance for Interoperability. (2007, June). *IFC2x3 TC1.* Retrieved May 4, 2009, from http://www.iai-tech.org

International Alliance for Interoperability. (2008, August 21). *ifcXML2x3.* Retrieved November 23, 2009, from http://www.iai-tech.org/products/ifc_specification/ifcxml-releases/ifcxml2x3-release/summary

International Energy Agency. (2006). Executive Summary. In *Light's Labour's Lost – Policies for Energy-efficient Lighting* (p. 25). Paris, France: OECD.

Jensen, H. W. (1995). Importance Driven Path Tracing using the Photon Map. *Rendering Techniques '95* (pp. 326-335). Dublin, Ireland: Springer .

Jensen, H. W. (2001). *Realistic Image Synthesis Using Photon Mapping.* Natick, Massachusetts: A K Peters.

Kajiya, J. T. (1986). The Rendering Equation. *ACM SIGGRAPH Computer Graphics, Vol.20, No. 4* , 143-150.

Kopylov, E. A., Khodulev, A. B., & Volevich, V. L. (1998). *Accurate Lighting Simulation in Computer Graphics Software.* Moscow: Keldysh Institute of Applied Mathematics.

Lam, K. P., Huang, Y. C., & Zhai, C. Q. (2004). *Energy Modeling Tools Assessment For Early Conceptual Design. Final Report to Northwest Energy Efficiency Alliance (Contract 10026).* Pittsburgh: Center for Building Performance and Diagnostics, Carnegie Mellon University.

Lam, K. P., Loftness, V., Hartkopf, V., Huang, Y. C., Zhai, C., & Bing, D. (30 November 2007). *Integrated Concurrent Design of High Efficiency Commercial Buildings. Final research report prepared for United Technologies Research Center.* Pittsburgh: Center for Building Performance and Diagnostics, Carnegie Mellon University.

Landman, M. (2005, December 23). *The Green Evolution: Tipping Towards the Mainstream.* Retrieved January 31, 2008, from Environmental Design + Construction: http://www.edcmag.com/CDA/Archives/c44c73b231d98010VgnVCM100000f932a8c0

Larson, G. W., & Shakespeare, R. (2003). *Rendering with Radiance - The Art and Science of Lighting Visualization. Revised Edition.* Davis: Space & Light.

Larson, G. W., Rushmeier, H., & Piatko, C. (1997). A Visibility Matching Tone Reproduction Operator for High Dynamic Range Scenes. *IEEE Transactions on Visualization and Computer Graphics, Vol. 3, No. 4* , 291-305.

Lawrence Berkeley National Laboratory. (2008). EnergyPlus Version 3.0.0 [Software].

Lawrence Berkeley National Laboratory. (2008). Radiance Version 3.9 [Software].

Maamari, F. (2006). *CIE 171:2006 Test Cases to Assess the Accuracy of Lighting Computer Programs.* Commission Internationale de L'Eclairage.

MacDonald, J. D., & Booth, K. S. (1990). Heuristics for ray tracing using space subdivision. *Visual Computer, Vol. 6, No. 3* , 153-166.

McGraw Hill. (2007, October 24). *Interoperability in the Construction Industry SmartMarket Report.* Retrieved November 30, 2007, from http://construction.ecnext.com/coms2/summary_0249-259123_ITM

Minnesota Planning. (2002, January). *Return on Investment: High Performance Buildings (Perspectives Series).* Retrieved May 4, 2009, from Minnesota Department of Administration: http://www.mnplan.state.mn.us/pdf/2002/BuildingsReport.pdf

National Institute of Building Sciences. (2008, August 07). *Whole Building Design Guide.* Retrieved January 31, 2008, from WBDG - Whole Building Design Guide: www.wbdg.org

National Science and Technology Council Committee on Technology. (2008). *Federal Research and Development Agenda for Net-Zero Energy, High-Performance Green Buildings.*

Ng, E., Lam, K. P., & Nagakura, T. (2001). Advanced Lighting Simulation in Architectural Design in the Tropics. *Automation in Construction, Vol. 10, No. 3* , 365-379.

Nicodemus, F. E., Richmond, J. C., Hsia, J. J., Ginsberg, I. W., & Limperis, T. (1977). Geometric Considerations and Nomenclature for Reflectance. *Monograph 161.* National Bureau of Standards (US).

Prowler, D. (2008, 8 7). *Whole Building Design*. Retrieved 8 31, 2009, from Whole Building Design Guide: http://www.wbdg.org/wbdg_approach.php

Reinhart, C. F., & Herkel, S. (2000). The simulation of annual daylight illuminance distributions- A state of the art comparison of six RADIANCE based methods. *Energy & Buildings, 32:2* , 167-187.

Roy, G. (2000). *A Comparative Study of Lighting Simulation Packages Suitable for use in Architecture Design.* Perth: Murdoch University.

Ruppertsberg, A. I., & Bloj, M. (2006). Rendering complex scenes for psychophysics using RADIANCE: How accurate can you get? *Journal of te Optical Society of America, Vol 23, No. 4* , 759-768.

Schregle, R. (2005). *Daylight Simulation with Photon Maps, Dr. Ing. Thesis.* Saarbrücken: Universität des Saarlandes.

Teghtsoonian, R. (1971). On the exponents in Stevens' law and the constant in Ekman's law. *Psychological Review, Vol 78(1)* , 71-80.

The American Society of Heating, Refrigerating and Air-Conditioning Engineers. (1999). *Standard 90.1-1999 (I-P Edition) Energy Standard for Buildings Except Low-Rise Residential Buildings.* Atlanta: ASHRAE.

U.S. Department of Energy. (2009). *2008 Buildings Energy Data Book.*

U.S. Department of Energy. (2001). *Greening Federal Facilites - An Energy, Environmental, and Economic Resource Guide for Federal Facility Managers and Designers (Second Edition).* Vermont: BuildingGreen Inc.

U.S. Green Building Council. (2005). *Green Building Rating System for New Construction and Major Renovations (LEED-NC), Version 2.5.*

Ubbelohde, M. S. (1998). Comparative evaluation of four daylighting software programs. *Proceedings of ACEE Summer Study on Energy Efficiency in Buildings.*

US Green Building Council. (2002). *National Trends and Prospects for High Performance Green Buildings.* Washington DC: US Green Building Council.

Vladimir, B. (2001). Acquisition of Building Geometry in the Simulation of Energy Performance. *Proceedings of Building Simulation '01* (pp. 305-312). Rio de Janerio: IBPSA.

W3C. (2000, October 6). *Extensible Markup Language (XML) 1.0 (Second Edition).* Retrieved November 23, 2009, from http://www.w3.org/TR/2000/REC-xml-20001006

Walter, B., Fernandez, S., Abree, A., Bala, K., Donikian, M., & Greenberg, D. P. (2005). Lightcuts: A Scalable Approach to Illumination. *ACM Transactions on Graphics, 24 (3)* , 1098-1107.

Ward, G. J. (1992). Measuring and Modeling Anisotropic Reflection. *Computer Graphics, Vol. 26, No. 2* , 265-272.

Ward, G. J. (1994). The RADIANCE Lighting Simulation and Rendering System. *Computer Graphics Proceedings, Annual Conference Series*, (pp. 459-472).

Ward, G. (1991). Real Pixels. In J. Arvo, *Graphic Gems II* (pp. 80-83). Cambridge: AP Professional.

Witte, M. J., Henninger, R. H., & Glazer, J. (2001). Testing and Validation of a New Building Energy Simulation Program. *Proceedings of Building Simulation 2001.* Rio de Janeiro.

Wong, N. H., Lam, K. P., & Feriadi, H. (1999). The use of performance based simulation tools for building and evaluation - a Singapore perspective. *Building and Environment, Vol.35, No. 8* , 709-736.