Iterative Refinement Through Simulation

Exploring trade-offs between speed and accuracy

Patrick Janssen[1], Vignesh Kaushik[2]
National University of Singapore
[1]patrick@janssen.name, [2]vigneshkaushik@gmail.com

Abstract. *Performance-based design approaches typically use iterative simulation as a way of exploring design variants. For such approaches, the speed of execution of the simulations is critical to enabling a fluid and interactive design process. This research proposes an iterative simulation design method where simulations are configured to run in two modes: in fast mode, simulations produce less accurate results but due to their speed can be applied successfully within an iterative refinement process; in slow mode, the simulations produce more accurate results that can be used to verify the performance improvements achieved using the iterative refinement process. A case study is presented where the proposed method is used to explore performance improvements related to levels of incident illuminance and incident irradiance on windows.*
Keywords. *Iterative; design; refinement; simulation; Radiance.*

INTRODUCTION

Visual Dataflow Modelling (VDM) (Janssen and Chen 2011) has becoming increasingly popular within the design community, as it can accelerate the iterative design process, thereby allowing larger numbers of design possibilities to be explored. Modelling in a VDM system consists of creating dataflow networks using nodes and links, where nodes can be thought of as functions that perform actions, and links connect the output of one function to the input of another function. VDM is now also becoming an important tool in performance-based design approaches, since it may potentially enable designers to explore and refine design possibilities through an iterative process of parametric variation coupled with performance simulation (Shea et. al. 2005, Coenders 2007, Lagios et. al 2010, Toth et. al. 2011, Janssen et. al. 2011).

In order for the process of iterative refinement to be effective, it is critical to set appropriate trade-offs between simulation speed and simulation accuracy. In order for simulations to be used fluidly and interactively, execution time must be kept to a minimum. However, the accuracy of the simulation is often inversely related to the speed of execution. Fast simulations produce low-accuracy results, while slow simulations produce high-accuracy results.

This paper proposes an iterative simulation design method that overcomes this problem by calibrating simulations to run both in a fast and less accurate mode and in a slow and more accurate mode. The fast mode simulations are used to enable designers to apply iterative refinement in a fluid and interactive manner, while the slow mode simulations are used to verify the performance improvements achieved using the iterative refinement process.

In order to demonstrate the proposed method, a case-study experiment has been conducted, where

the method is used to explore design variants for a large residential project consisting of over a thousand units. Design variants are evaluated based on visible daylight and radiant heat (including sunlight) incident on the surface of the windows of residential units.

SIMULATION NODES

Visible daylight is measured as *illuminance*, which is the visible light incident on the surface, and is measured in Lumens/m^2 or Lux. Radiant heat is measured as *irradiance*, which is the electromagnetic radiation incident on the surface, measured in Watts/m^2. Both illuminance and irradiance are calculated using the simulation program Radiance [1].

Radiance simulations

Radiance is a collection of programs that perform a variety of related tasks. The main input file for Radiance is the RAD file that describes the scene to be simulated. Given a RAD file, the first step is to convert this into a different file format called an *octree*, using a program called *oconv*. Using this octree file as input, the user can specify sensor points in the model and then use the *rtrace* program to measure illuminance or irradiance at these points.

When the octree is generated using the *oconv* program, the radiance description of the sky dome can be included. Different skies need to be generated for the illuminance and irradiance simulations. For the illuminance simulation, standard CIE overcast sky is required, as this is the worst case scenario for calculating daylighting. This sky is not affected by the position of the sun, and as a result it is time independent. The illuminance that is calculated will then represent the actual Lux value for the worst case scenario.

For the irradiance simulation, rather than focusing on the worst case, a cumulative approach is needed whereby the irradiance incident on a particular point throughout the year is added up. In addition, irradiance is of course time dependent as it is affected by the position of the sun. One option for calculating cumulative irradiance would be to generate skies for multiple points in time and then to run multiple simulations. However, a more efficient approach is to create a cumulative annual sky from a climate file (Robinson and Stone 2004). The irradiance results from a single simulation run using such a cumulative sky would then represent cumulative irradiance for the whole year.

For generating the standard CIE overcast sky for the illuminance simulations, Radiance includes a program called *gensky*[2]. For generating the cumulative annual sky, a program called *GenCumulativeSky*[3] is used. This program produces cumulative annual skies in Radiance format from EnergyPlus weather files [4]. The sky is discretized into 145 patches using a method devised by Tregenza (1987) and the Perez luminance/radiance distribution model (Perez et. al. 1993) is used to determine the radiance of each patch, according to the information from the climate file. Figure 1 shows both the standard CIE overcast sky and the cumulative annual sky used in this research.

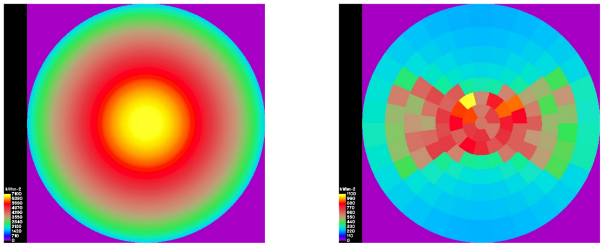


Figure 1
Falsecolor image of the standard CIE overcast sky generated by gensky *(left) and the cumulative annual sky generated by GenCumulativeSky (right) using the EnergyPlus weather file for Singapore.*

Radiance VDM nodes

In order to support a user-friendly integration of Radiance into the design workflow, VDM nodes were created for an advanced procedural modelling system called SideFX Houdini [5]. These nodes link Houdini with the various Radiance programs and the *GenCumulativeSky*program. For more information on the development of these nodes, see Janssen et. al. (2011). For this research, the nodes were further developed to allow users to create a sky via three methods: by using the *gensky* program, by using the *GenCumulativeSky* program, or by loading a sky description file.

The main simulation node executing *oconv* and *rtrace* has two inputs: one for the model geometry and one for sensor grids. The first input is for inputting the geometry. The node will translate each Houdini polygon to a Radiance primitive of type polygon. The polygons expected to have custom attributes to define the material, and the node will extract the values of these attributes when generating the Radiance input file.

The second input of the Radiance node is for inputting sensor grids. When the *rtrace* simulation is run, the simulation results will be copied to the sensor points within Houdini as attributes. This then means that the results from an *rtrace* simulation can be graphically displayed inside Houdini, using coloured surfaces.

Simulation parameters

In terms of speed, *rtrace* will tend to take significantly more time to execute than the other programs such as *oconv*, *gensky*, and *GenCumulativeSky*. The settings for the *rtrace* parameters are therefore critical when it comes to the trade-off between speed and accuracy.

The Houdini node provides parameters for specifying certain key *rtrace* parameters relating to ambient lighting, namely [6]:

- Ambient bounces (ab): the maximum number of diffuse bounces computed by the indirect calculation. A value of zero implies no indirect calculation
- Ambient accuracy (aa): the approximate error from indirect illuminance interpolation. A value of zero implies no interpolation
- Ambient resolution (ar): The maximum density of ambient values used in interpolation. Errors will start to increase on surfaces spaced closer than the scene size divided by the ambient resolution
- Ambient divisions (ad): The error in the Monte Carlo calculation of indirect illuminance will be inversely proportional to the square root of this number. A value of zero implies no indirect calculation
- Ambient super-samples (as): Super-samples are applied only to the ambient divisions which show a significant change

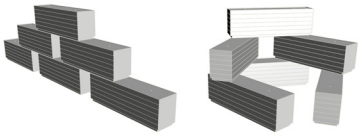
A detailed explanation of these parameters is beyond the scope of this paper (see the Radiance documentation for more information). However, it should be noted that perhaps the most important parameter is the ambient bounces. The number of bounces may vary from 0 to 8, with higher number of bounces producing more accurate results but also in much higher computation times. Since the sky is modelled as 'glow', it will only take part in Radiance's indirect lighting calculation, which is only performed when ambient bounces is set to 1 or more. Since the only light is coming from the sky, then a value of 1 is equivalent to calculating only direct and diffuse light, but ignoring any reflected indirect light.

For the ambient accuracy parameter, a lower value produces more accurate results with slower execution times. However, note that if no reflected indirect lighting is calculated (as is the case when the only light is coming from the sky dome and the ambient bounces parameter is set to 1), then this parameter can be set to 0, as it will not have any effect.

For the final three parameters, higher values will generally produce more accurate results but with slower execution times.

CASE STUDY EXPERIMENT

The case study experiment will focus on the Interlace, a large residential project designed by OMA [7] and currently under construction in Singapore. The design consists of thirty-one apartment blocks, each six stories tall. The blocks are stacked in an interlocking brick pattern, with voids between the bricks. Each stack of blocks is rotated around a set of vertical axes, thereby creating a complex interlocking configuration. An example is shown in Figure 2, where 6 blocks are stacked and rotated to form a hexagonal configuration.



Each block is approximately 70 meters long by 16.5 meters wide, with two vertical axes of rotation spaced 45 meters apart. The axes of rotation coincide with the location of the vertical cores of the building, thereby allowing for a single vertical core to connect blocks at different levels. The blocks are almost totally glazed, with large windows on all four facades. In addition, blocks also have a series of balconies, both projecting out from the facade and inset into the facade. A typical floor plan is shown in Figure 3.

The OMA design has stacked the 31 blocks into 22 stacks of varying height, and has then rotated the stacks into a hexagonal pattern constrained within the site boundaries. At the highest point, the blocks are stacked four high.

Design exploration task

For this research, an exploration task has been defined, in which the designer is required to minimize the number of windows receiving either low illuminance or high irradiance. The designer will carry out this exploration task via a process of iterative refinement, whereby a parametric model is built and the parameters in the model are gradually adjusted in order to try and improve performance. Each iteration of parametric adjustment by the designer is followed by a simulation of design performance, and if performance improves then the parametric changes are kept. Using this approach the designer may gradually be able to improve performance of the design.

For this task, the parametric changes that can be made by the designer have been constrained to the rotation of the blocks and the addition of sun shades. Block rotation a change that affects global configuration, while sun shading is seen as a change that affects only local configuration. In order to constrain the task, other possible changes, such as the stacking of the blocks and the position and size of balconies were not considered. However, it is noted that the iterative approach used in this research could also be expanded to include such parameters.

In order to test the iterative approach, a Houdini model of the design was built that included all significant exterior features including walls, windows, inset balconies and protruding balconies. On the interior, most of the detail was omitted and only unit walls were included. The resulting model had a total of approximately 47.5 thousand polygons, of

Figure 2
The process of rotating the brick pattern. The diagram on the left shows 6 blocks arranged in a straight line, while the diagram on the right shows the same six blocks folded into a hexagonal pattern.

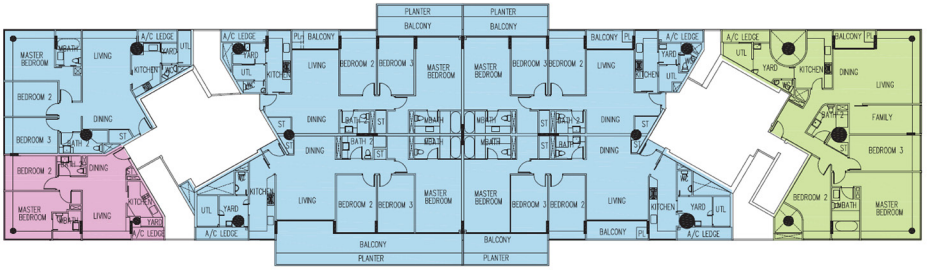


Figure 3
A typical floor plan [8].

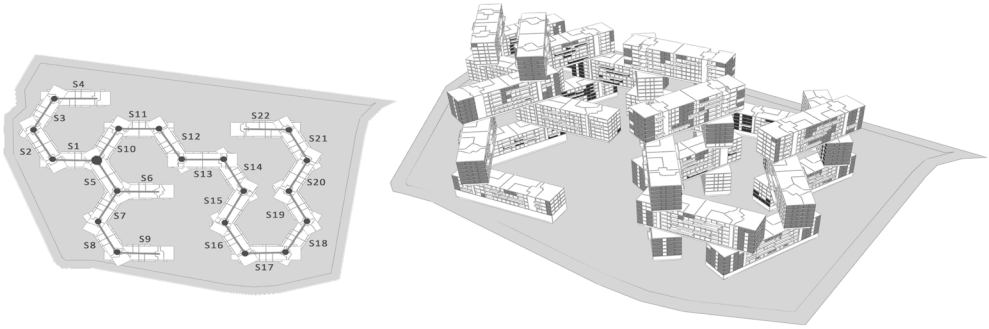


Figure 4
The original design. The plan on the left shows the root node and the branching structure. The model on the right shows windows with low illuminance in dark grey, and windows with high irradiance in light grey.

which about 7800 were windows. These windows were grouped into four types: living room windows, bedroom windows, kitchen windows, and utility windows. For the performance exploration, it was decided to focus on the living rooms and bedroom windows only, which totalled 5250 windows. The illuminance and irradiance incident on each window was measured at just one point in the centre of the window. Therefore, for each iteration, illuminance and irradiance was to be measured at 5250 points in the model.

For the exploration task, target thresholds were set for both illuminance and irradiance. Windows falling either below the illuminance threshold or above the irradiance threshold were considered to be undesirable, and therefore in need of improvement. The aim of the exploration task was to reduce the total number of undesirable windows. These thresholds were mainly used as a simple way of summarizing relative performance, so that the designer was able to quickly assess whether improvements has been made.

Parameterisation of the model

In order to allow the designer to fluidly and interactively make changes to the rotation angles of the stacks of blocks, the blocks need to be parametrically linked. Looking at the arrangement of the blocks in plan in Figure 4, it is evident that the configuration is actually a branching hierarchical structure, with a central root and three branches. This type of branching structure can be modelled within animation tools such as Houdini using objects that have parent-child relationships. In the plan in Figure 4, the root node is indicated by the larger dot and is the parent of three block stacks: s1, s5 and s10. Each of these three stacked blocks is the start of one branch. The parent-child linking relationship means that any transformation applied to an object will automatically also be applied to all the descendants. The designer can therefore freely explore different rotation combinations without having to worry about the stacked blocks becoming disconnected.

Iterative simulation design method

The key step in the iterative simulation design method was the executions of the simulations. Calculating the illuminance and irradiance at a high level of accuracy can be very time consuming, and therefore very disruptive for the designer.

For obtaining accurate results, the following Radiance ambient settings were used: ab=4, aa=0.15, ar=2048, ad=516, and as=516. Using these settings, the illuminance simulation took 8 hours and 30 minutes and the irradiance simulation took 13 hours 50 minutes making a total of 22 hours and 20 minutes. The computer being used for running the simulations was a typical office computer: a 2.4GHz dual-core processor with 8GB RAM running 64 bit Windows.

The simulation results showed that within the existing design, a significant portion of windows had either low illuminance or high irradiance. The illuminance and irradiance patterns on the facade were also seen to be very varied and hard to predict due the complex massing of the building, and also due the effects of protruding balconies shading windows below. The iterative simulation design method was therefore deemed to be appropriate for exploring options with fewer undesirable windows. However, due to the excessive simulation time, an iterative simulation design method was developed where each simulation was configured to run in fast mode and in slow mode. The aim was to reduce the total simulation time of the fast version to below two minutes, but to ensure that the results from the fast and the slow mode simulations still correlated reasonably well. This would then allow the fast simulation to be used as a driver for the exploration process.

The iterative simulation design method was divided into three main phases: calibration, iteration, and verification. In the calibration phase, the fast mode simulations are set up and configured in order to ensure that appropriate trade-offs are achieved between speed and accuracy. In the iteration phase, the fast mode simulations are used within the iterative refinement process in order to explore design variants with improved performance. Finally, in the verification stage, both the initial design and the final design from the iterative process are evaluated using the slow mode simulations in order to verify the performance improvements. The three phases of the iterative simulation design method are shown in Figure 5.

The calibration phase

For the calibration phase, a series of Radiance simulations were executed with parameter settings that favoured speed over accuracy. In all cases, the ambient bounces parameter was set to 1 and the ambient accuracy parameter was set to 0. This therefore meant that no indirect reflections were calculated which significantly reduced the execution time. For each of these simulations, Microsoft Excel was then used to plot the trend-line between the fast and slow mode simulation results, and to calculate the R^2 correlation coefficient (or the coefficient of determination). Table 1 shows the results for these experiments.

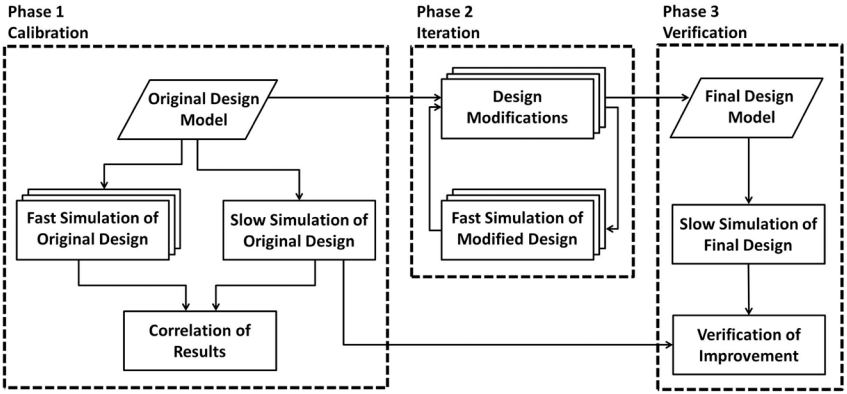


Figure 5
The three phases of the iterative simulation design method.

Table 1
Table showing the execution time (T) and R^2 correlation for a range of different ambient light settings for the Radiance rtrace simulation.

Radiance rtrace ambient settings	Illuminance		Irradiance	
	T	R2	T	R2
ab=1, aa=0, ar=2048, ad=512, as=512	92s	0.8892	88s	0.8841
ab=1, aa=0, ar=1024, ad=256, as=256	49s	0.8892	53s	0.8839
ab=1, aa=0, ar=512, ad=128, as=128	32s	0.8875	36s	0.8796

Based on the execution time and R^2 correlation results, it was decided that for both the fast illuminance simulation and the fast irradiance simulations, the second set of settings from table 1 would be used. These settings allow the simulations to be executed in under 1 minute each, and also maintain an R^2 correlation of close to 0.9.

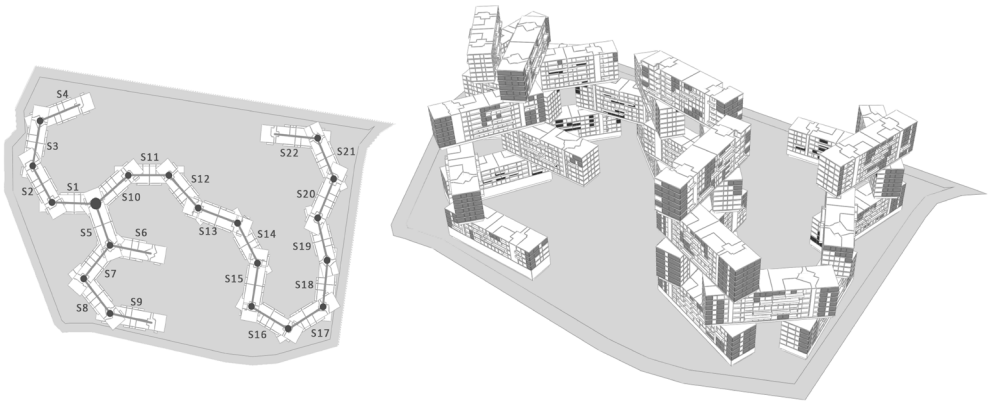
The final step in setting up the fast simulations was to map the results from the fast simulation using the linear trend-line equation. Microsoft Excel was used to obtain the linear trend-line equation, which was then transferred back to Houdini, where it was used to map the results from the fast simulation. This option for mapping the simulation results was provided as part of the Houdini node. In effect, this mapping of the fast simulation results adjusts the trend line so that it passes through the graph origin at 45 degrees.

The iteration phase

Within the Houdini environment, the total number of undesirable windows for both illuminance and irradiance were continuously displayed to the designer as both numeric totals, and as coloured polygons within the three-dimensional model. Once the designer had made a set of changes to the model, they were then able to trigger the simulations to re-execute. After two minutes, once the simulations completed executing, both the numeric totals and the coloured polygons would be automatically updated, thereby giving fast feedback to the designer as to whether their changes resulted in better performance.

The exploration process was set up as a two stage process. In the first stage, the rotation parameters were iteratively explored. For each iteration, the designer would identify a particular cluster of windows with low illuminance, and would then make a

Figure 6
The design modified in order to reduce the number of windows with low illuminance. The plan on the left shows how the branching structure has been modified to try and increase the openness between the branches. The model on the right shows windows with low illuminance in dark grey, and windows with high irradiance in light grey.



small number of changes in order to try to reduce the obstructions for those windows. In some cases, such changes would indeed improve the situation, but in other cases, the changes would cause deterioration in performance in some other part of the design. At this stage, the focus was on reducing the number of windows with low illuminance, as this was deemed to be a more challenging task. However, the designer also kept a check on the number of windows with high irradiance, since changes that improved illuminance often also resulted in higher levels of irradiance. The final design for stage 1 is shown in Figure 6.

In the second stage, the best solution from the first stage was selected and the addition of solar shading devices was then explored with the aim of reducing the number of windows with high irradiance. For the rotation parameters, the changes were applied manually, since there were only 22 set of stacked blocks. However, for the windows, the manual approach could not be used since there were thousands of windows. An automated approach was therefore created within Houdini whereby shading devices were parametrically generated for the windows with high levels of irradiance. The depth of the shading devices was varied in relation to level of irradiance on the window. In this case, the iterative process was used to explore the relationships between the depth of the shading devices and the level of irradiance on the window. As with stage 1, the designer also kept a check on the number of windows with low illuminance, since the addition of solar shading devices reduced illuminance levels in some cases.

It was found that in the first stage, the reduction of the number of windows with low illuminance was difficult to achieve. The number of low illuminance windows was reduced through an iterative refinement process consisting of 18 iterative steps. In the second stage, the windows with high irradiance were more easily solved using additional sun shading devices. During this stage, the number of high irradiance windows was reduced in 6 iterative steps.

The verification phase

In order to verify that performance had indeed been improved, the initial design and the final design were evaluated using the slow mode simulations and the results were compared. Note that the goal of this verification was not to compare the results from the fast mode simulations with those from the slow mode simulations, but rather to measure the actual performance improvements that were achieved through the iterative refinement process. Despite good R^2 correlations of close to 0.9, the results from the fast mode simulations could not be used as an objective measure of performance. Instead, the fast simulation modes were used only as a way of measuring relative performance, and within the iterative phase were used as a driver for the exploration process.

The slow mode simulation results show that the total number of windows with low illuminance and high irradiance have been reduced by 8% and 32% respectively. This confirms that the performance was successfully improved using the iterative simulation design method.

CONCLUSIONS

This research aimed to explore the trade-off between speed and accuracy when applying iterative simulation approaches to complex designs where the size of the digital models typically becomes large, and as a result execution times for simulations may become prohibitively slow.

This research has explored an approach in which simulations are run in two modes: fast mode and slow mode. An iterative simulation design method has been developed consisting of three phases: in the first phase, fast mode simulations are calibrated by setting appropriate trade-offs between speed and accuracy; in the second phase, the fast mode simulations are used to iteratively refine the design in response to performance feedback; lastly, in the third phase, the performance improvements achieved through the iterative refinement process are verified. The application of the proposed meth-

od to a complex case study of a large residential design demonstrates the feasibility of the approach.

Future research will focus on further exploring how the proposed approach can be applied to a wider range of simulation tools, including structural simulations and energy simulations.

REFERENCES

Coenders, JL 2007, Interfacing between parametric associative and structural software.In *Proceedings of the 4th International Conference on Structural and Construction Engineering*, Melbourne, Australia, 26–28 September.

Janssen, PHT, Chen, KW and Basol, C 2011, Iterative Virtual Prototyping: Performance Based Design Exploration. In *Proceedings of The International Conference on Education and research in Computer Aided Architectural Design in Europe (eCAADe '11),* pp. 253–260.

Janssen, PHT and Chen, KW 2011, Visual Dataflow Modelling: A Comparison of Three Systems, in *Proceedings of the CAAD Futures '11*, pp. 801–816.

Kolarevic, B and Malkawi, A 2005, Operative Performativity (panel discussion), in *Performative Architecture: Beyond Instrumentality*, New York :Spon Press, 2005, pp.239–246.

Lagios, K, Niemasz, J, and Reinhart, CF 2010, Animated Building Performance Simulation (ABPS) – Linking Rhinoceros/Grasshopper with Radiance/Daysim, in *Proceedings of SimBuild 2010*, New York City, August 2010

Robinson, D and Stone, A 2004, Irradiation modelling made simple: the cumulative sky approach and its applications, In*Plea2004 – The 21st Conference on passive and low Energy Architecture*, Eindhoven, The Netherlands.

Tregenza, P 1987, Subdivision of the Sky Hemisphere for Luminance Measurements, *Lighting Research and Technology*, Vol 19, pp. 13–14.

Toth, B, Salim, F, Frazer, J, Drogemuller, R, Burry, J, and Burry, M 2011, Energy-oriented design tools for collaboration in the cloud. *International Journal of Architectural Computing*, 4(9): 339–359.

Shea, K, Aish, R, and Gourtovaia, M 2005, Towards integrated performance–driven generative design tools. *Automation in Construction*. March 2005, 14(2): 253–264.

[1] http://radsite.lbl.gov/radiance/

[2] http://radsite.lbl.gov/radiance/man_html/gensky.1.html

[3] http://diva4rhino.com/

[4] http://apps1.eere.energy.gov/buildings/energyplus/cfm/weather_data.cfm

[5] http://www.sidefx.com/

[6] http://radsite.lbl.gov/radiance/man_html/rtrace.1.html

[7] http://oma.eu/projects/2009/the-interlace

[8] http://www.theinterlace.com.sg/