J. Zuo, L. Daniel, V. Soebarto (eds.), *Fifty years later: Revisiting the role of architectural science in design and practice: 50th International Conference of the Architectural Science Association 2016,* pp. 417–426.

Games technology and building performance analysis

Andrew Marsh[1] and Sid Thoo[2]

[1] *PerformativeDesign.com, Douglas, Isle of Man*
andrew.marsh@performativedesign.com

[2] *Curtin University, Perth, Australia*
s.thoo@curtin.edu.au

Abstract: Modern games technology has advanced rapidly in the past few years. This is true of both open source and commercial games engines, both of which provide an infrastructure for the real-time visualisation of complex geometry with dynamic lighting, shadowing and texture effects, together with support for sound, animation and responsive user interaction. Games engines are extremely flexible – just as capable of rendering flight simulations, battleship flotillas, urban car chases or the interiors of spaceships and dungeons – all whilst overlaying detailed user interface elements and dynamic contextual information. This paper describes the first stage of a research project that aims to utilise this new technology for better representation of building performance analysis information and concepts. The initial work of building the frameworks required for visualising and interacting with this kind of data is presented as well as how gamification concepts have been used in the development of some preliminary online educational tools used to demonstrate and test it.

Keywords: Games engine; gamification, visualisation; simulation; analysis.

1. Introduction

Gamification refers to the application of games design principles and techniques to other contexts that are not typically games-related. This definition is quite broad, but an important point is that it does not always mean turning something into a game. In many contexts it can simply mean exploiting game elements that stimulate engagement, motivation, playfulness and a sense of achievement. In the context of the research presented here, gamification is used to mean the creation of highly interactive application environments in which quantifiable performance goals and progress towards them are elevated to the forefront and the means of further progress or achievement are obvious from the controls and interactions made available to the user.

This research is in its formative stages as there is significant initial work required to develop even the most basic and experimental framework for visualising and interacting with building geometry and performance analysis data. The term 'experimental framework' is important here as there is no obvious blueprint for what this should be or how it should work. Popular games vary markedly in their visual interface design and the types of user interaction and manipulations they offer, even though they are

generally considered to be visually appealing and quick to learn (Chou, 2015). CAD, BIM and 3D modeling applications have more similarities in their visual interfaces and user interaction, but are typically more complex and have much steeper learning curves. Thus, developing ways to bridge these two paradigms is expected to be an iterative research exercise involving significant trial and error.

This paper outlines the rationale behind the first set of these iterations and introduces some educational tools that have been created to demonstrate and evaluate parts of this work. As such, it touches only on the first of what will be a three-phase research project. Only Phase 1 is currently being worked on. Phases 2 and 3 will commence when the infrastructure is sufficiently developed to support them.

The three phases of the overall project have the following aims:

1. To develop core frameworks and control elements that allow building geometry and analysis data to be visualised and interacted with from within a range of games engines and platforms,
2. To objectively assess the most effective and informative visualisation and interaction techniques for the many different types of building simulation and analysis data available, and
3. To develop more advanced frameworks for the gamification of common building performance analysis processes and their integration into architectural design workflows and best practices.

2. Approach

Recognising the iterative nature of the first phase, the approach taken has been to create a series of educational tools that can serve as both realistic short-term development goals and a means of investigating, demonstrating and assessing different aspects of the problem. The intention is that these tools gradually increase in complexity and scope, with the utility and robustness of the shared code base gradually accumulating with each one.

The first of these tools were used as a means of mastering the various technologies and techniques involved and to gain implementation experience. However, the most recent tools have begun to include more sophisticated visualisations, user interactions and some preliminary gamification techniques. Three of these tools are discussed here, each shown in Figure 1 – the first focusing on solar position, the second on the relationship between location and shadows and the third on room daylight distribution.

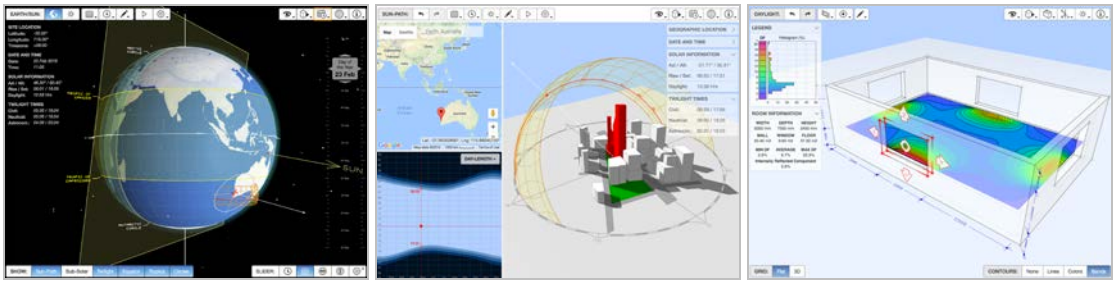

Figure 1: Screenshots from three of the educational tools developed as part of this research.

In addition to their role in the development process, the aim of these educational tools is to illustrate and demonstrate specific aspects of building performance analysis in a way that is engaging and that encourages deliberate investigative play. The game elements identified as being key to achieving this are highly interactive analysis models with dynamic visual feedback and low latency between user action and model reaction. This basically means being able to physically manipulate the main parameters that govern a model's behavior and have the visual analysis update in as close to real time as possible.

3. Why games technology

There are a number of characteristics that make games technology interesting from an architectural science perspective. The primary one has to be dynamic interactivity. It is certainly possible to use 3D modelling or rendering applications to create pre-rendered images and animations that show building geometry overlaid with simulation and analysis data. However, it takes time to set up and render new images so the process of iterating, investigating and even changing direction as each visualisation feeds back new information into the design process can be very slow.

Educationally, reducing the duration between model change and analytical feedback can greatly assist the development of a more intuitive understanding of causal processes (Lowe and Schnotz, 2008) such as building performance. The ideal is obviously instantaneous feedback as the cause and effect relationships are immediately obvious. Not all physical processes can be viably simulated in real-time, but some can and games technology has the potential to wrap those into very useable and useful educational tools.

For those that can't, games technology alone is no solution. However, providing tools to make exporting and re-exporting models to an environment in which they can be freely navigated or examined as the viewer desires is the first step. Providing simple and intuitive ways to select, control and compare different analysis results and how they are mapped over the model is perhaps a further step. The final step is devising insightful visualisations and animated data displays that can be navigated or examined just as freely as the model itself.

The following is a brief discussion of the features of games engines, how they contribute to dynamic interactivity and why they provide a proven base within which to take these steps.

3.1. Rendering speed

Games engines are typically engineered for very fast rendering of complex and often dynamic geometry at high frame rates. The primary reason for this is that most players prefer detailed and visually rich game environments that they can move freely around in without noticeable flicker. Also, they prefer actions and other characters within the game to be animated smoothly and realistically, again without noticeable flicker. Serious gamers will invest significant amounts in high quality hardware to achieve this and research has shown that this preference extends well beyond just gamers (Gonzales, 1996).

Compared to the complex dynamic environments of many modern games, the requirements for architectural and analytical visualisation are relatively modest. For architecture, this technology allows even highly detailed building models with extensive texturing and shading to be interactively rotated or navigated around very smoothly. It also allows for rich animation of camera views, building geometry and even overlays such as data points, analysis grids and volumetric information. This contrasts significantly with the focus in almost all CAD and BIM tools on static plan, section and elevation projections as well as stored camera views.

Research (Bederson, Boltman, 1999) suggests that interactive animation when navigating a 3D environment or spatial dataset helps users maintain their sense of positional constancy without having to rebuild their understanding of where things are again after a view change or position shift. Even for two dimensional data, work by Microsoft Research (Heer, Robertson, 2007) shows that animating between different chart projections or graphical representations allows users to more easily track both individual data values and the complex relationships that exist between different values.

When working with large spatial datasets that can also change over time, the ability to maintain positional constancy and track complex relationships as views, values and scales change is critical to being able to both assess and understand such datasets. The rendering speed of games engines enables this, as well as the potential to use a wide range of new and dynamic data visualisation techniques. The raw graphics power of modern GPUs makes viewing large volumetric data possible even on a tablet or phone. Easily swiping or transitioning between multiple result sets in the same view also becomes possible, which makes better use of the ability of the human visual system to readily spot areas of subtle difference or slight change.

3.2. User interaction

Graphically intense games typically involve complex user interaction. Responsive and realistic action and reaction with low latency is often the key to how challenging a game feels and how much perceived skill or experience a player must develop in order to master it. Whilst CAD and BIM tools also involve complex user interaction, it is entirely different in nature and primarily aimed at facilitating data entry and the creation or editing of model geometry.

The benefit of using a games engine over a CAD or BIM tool is that pretty well all user-generated events and gestures from a wide range of input devices can be customised to almost any purpose or requirement. This includes touch and motion events as well as head and eye movements from sensors within a VR headset or glasses. This opens up significant opportunity for more involved and immersive interaction with both models and data, as well as the design of user interfaces that are highly customised to very specific tasks or types of analysis.

In their joint thesis, Fagerholt and Lorentzon (2009) researched game user interface design and introduced terms for the different types of interaction components they encountered based on how they related to both the player and the game environment. Whilst very games-oriented in definition, their concept of diegetic and non-diegetic interaction elements is still a very useful characterisation for some of the user interface elements being developed as part of this work.

Diegetic interface elements refer to parts of the 3D model that can be interacted with and manipulated within 3D model-space coordinates. Non-diegetic refers to 2D elements overlaid on top of a 3D model that are interacted with and manipulated within flat screen-space coordinates.

Figure 2 shows some of the interactive interface elements of each type developed as part of this research. These developed from extensive internal testing with touch, mouse and stylus inputs over a range of platforms and screen sizes. Phase 2 of this research will involve a more formal objective assessment of their utility and intuitiveness within different simulation and analysis processes, however the aim at Phase 1 is to explore the technology and techniques required and gain important experience with their implementation.

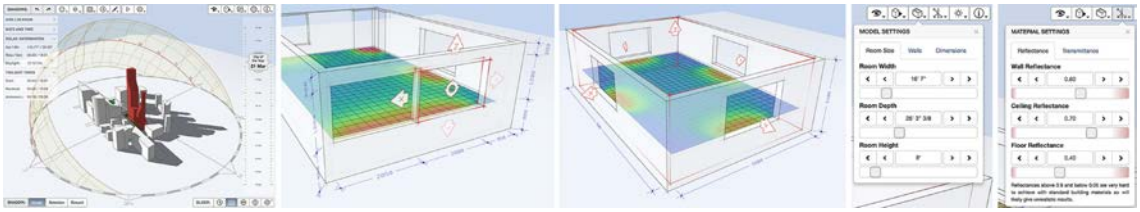

Figure 2: Designing interface elements for interactively manipulating models and analytical information, overlaid as both a heads-up-display and within the context of the 3D model.

3.3. Visual abstraction

The graphic styles of different games can vary quite markedly. Thus games engines need to be very flexible in both the types of geometry they can handle and how that geometry can be rendered. Whilst some CAD and BIM tools can be flexible in the types of geometry they can handle, they are often very inflexible in how it can be rendered and then included within the visible building model.

This flexibility in games engines comes mainly from providing direct access to the Graphics Processing Unit (GPU). This means being able to use custom low-level shader programs to control how individual objects within the scene will be rendered or to apply complex pre and post processing effects to the whole scene. This is not possible within CAD or BIM tools as they completely control the render process and the appearance of individual objects within a scene is governed entirely by the particular material parameters each tool provides.

The main benefit of custom shaders is that they allow for potentially unrealistic but useful visualization effects that might provide additional insight or understanding that a traditional photo-realistic render may not. The display of shadows is a particularly illustrative example of this.

3.3.1 Shaders and shadows

Most CAD and BIM tools can display the shadows cast by a building model and do a very good job of dealing with transparency and material effects. However, it is typically not possible to customize their shadow projections in any meaningful way; such as choosing a different colour, selecting specific surfaces to cast and/or receive shadows, or even differentiating the effects of only part of the geometry. However, the flexibility of games engines allows for a whole range of potentially innovative shadow visualisations, such as:

- Using custom shaders that can display multiple shadow maps created from different parts of the model or even generated at different times of the day or days of the year,
- Using multiple shadow maps to identify which geometric object is shadowing each pixel in the rendered image and then colouring it to highlight the effects of individual shapes in the model,
- Selecting the shallowest shadow map value at each pixel, making it possible to very effectively isolate the full extent of shadows cast and/or received by a particular object,
- Selecting the deepest shadow map value instead to see only the additional shadow created by a particular object when compared to a given base condition,

- Separating the shadow generation process entirely from model rendering, allowing the use of more complete or even completely different geometry to be used to create the shadows which can then be mapped over just a subset or even a different representation of the model, and
- Using a remote server to generate a highly detailed shadow map using the full building model and its surroundings, and then downloading that to a mobile phone or tablet device for mapping over a much simpler building model.

Figure 3 shows some examples of the types of shadow visualisations that custom shaders allow. These shadow images do appear similar to those already available using Ecotect (Marsh, 2003), however differentiating shadows with different colours and on specific surfaces is only possible within Ecotect's canvas-based '3D DESIGN' tab, not in the OpenGL-based 'VISUALISE' tab. In this work, all shadows are generated in OpenGL and WebGL so the core technology involved and the potential for even greater innovation and control represents a significant extension.

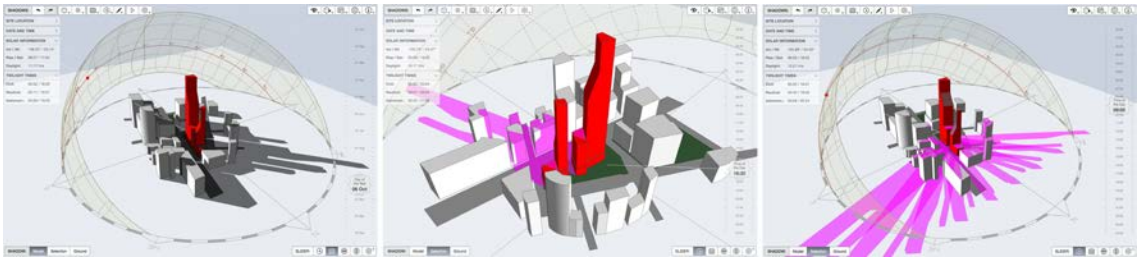

Figure 3: Examples of custom shadow visualisations. The left image shows a traditional shadow projection whilst the middle shows isolated shadows from just the two highlighted towers. The image on the right shows the same isolated shadows for each hour of the day from 9am to 5pm.

3.3.2 Access to the GPU

The creative potential of custom shaders cannot be overstated. Even a cursory browse through a website such as ShaderToy (Beautypi, 2016) will reveal all sorts of fractals, fluid dynamics and noise functions feeding into procedural textures and even highly detailed programmatic geometries. The computational capabilities of modern GPUs means that a whole range of physical effects that were previously only available in high-end rendering tools can now be simulated in real-time. Also, shaders can be used to interpolate between multiple vertex attributes or even morph an object between different geometric forms, allowing for smooth transitional animations and streaming dynamic data.

Having direct access to the GPU also means potentially using tools such as CUDA and OpenCL to perform complex analytical calculations as well as rendering. As modern GPUs are massively parallel processors, there are many building performance calculations that could potentially be performed much faster or even in real time. This is not quite as easy in browser-based games engines as there is no access to CUDA or OpenCL, and the new WebCL standard is not yet widely supported. However there are still ways to achieve something similar by co-opting standard WebGL shaders and obtaining results by extracting them from rendered textures (Github Inc. 2016). This is the technique that was used in the real-time GPU-based daylight analysis tool presented here.

4. Gamification

To stimulate engagement and deliberate investigative play in these initial educational tools, the gamification elements concentrated on most have been the generation of highly responsive visual feedback and an attempt to make the metrics of each analysis as prominent as possible within the user interface.

4.1 Responsive visual feedback

The ideal of instantaneous feedback requires significant optimisation of every aspect of the modelling, analysis and rendering pipelines. As an example, just to dynamically move a window within a wall and update the spatial daylighting requires the following steps:

- Check that the window is not overlapping another or moving outside the wall boundary,
- Update the window position and then regenerate, re-tessellate and update the mesh geometry of the wall and its apertures on the GPU,
- Recalculate dimension lines then regenerate and update their meshes on the GPU,
- Compute new daylight values at each grid point then update the colour and position of each vertex of the grid mesh on the GPU,
- If contours are displayed, recalculate each contour line or band boundary and again regenerate, re-tessellate and update their meshes on the GPU,
- Compute the binned value of the histogram/cumulative value chart and update its SVG path, and
- Update any textual data values visible within the user interface that may have changed.

The primary optimisations over which an application developer has most control is the computation of daylight values over the grid and the calculation of contours and dimensions. Using a games engine makes the process of updating geometry meshes on the GPU relatively trivial and, depending on individual implementations, also their regeneration and re-tessellation. However, this can still involve a lot of data transfer between the CPU and GPU so is not without its own time cost.

To achieve a refresh rate of 30 frames per second requires that all of the above steps be completed in around 33 milliseconds. Depending on the computing power available on the device running the application, this can place restrictive limitations on the resolution of the analysis grid and the complexity of the dynamic model. To overcome this, there is growing potential to utilise multiple threads on the CPU and the large number of parallel processing units available on the GPU.

As one of the aims of these educational tools is to provide experience in the implementation of new technologies, a custom shader has been developed for the computation of daylight values across a spatial grid. In its current form, this is based on the BRE Split Flux Method (Hopkinson, Petherbridge and Longmore, 1966) with a rectangular-shaped room and any number of simple frameless apertures. Though a simplified method (Kota and Haberl, 2009), a detailed comparative analysis of results from Radiance on the same simple room models with the same window layouts will soon be published and shows high correlation. This means that the method accurately captures the relative variations in daylight level when the room size or window layout is changed.

The main point at this stage of the research is not the accuracy or otherwise of the particular algorithms implemented in each tool. It is that these kinds of algorithms are implementable and that visualisation frameworks can be developed to allow for real-time interaction with them. Future work or

work by others can refine these algorithms or implement newer more appropriate methods within the frameworks being developed.

4.1.1 Accurate numerical input

Another issue highlighted by this work is the importance of accurate numerical input. Analysis algorithms require accurate numerical data for their key parameters in order to get accurate results. Yet at the same time, gamification requires the ability to easily and dynamically manipulate parameter values as a way of quickly exploring relationships within a model. Usability testing showed that invoking the onscreen keyboard on many mobile devices in order to enter a numeric value was disruptive to investigative play and that dragging a slider made accurate data entry too difficult, especially in a touch-based environment and for parameters that have a wide potential value range but which may be sensitive to changes within multiple decimal places.

The adopted solution has been to create a standard numerical input block that includes increment/decrement buttons, a formatted edit box and a customised dynamic slider, as shown in Figure 4. Large comparative changes can be done using the slider whilst exact known values can be entered using the edit box and system keyboard. The buttons allow for the value to be more slowly and deliberately changed in known increments. The slider needed customisation to significantly widen its track area as, in a number of browsers, the standard HTML5 range input slider track is quite small which touch-based control difficult.

Figure 4: Examples of the standard numerical input block developed to allow both interactive manipulation and accurate input of numeric parameter values.

4.2 Quantifiable performance goals

The solar position and location/shadow tools focused primarily on fast visual feedback whereas the daylight analysis tool provided the opportunity to prominently display statistical information derived from the calculated spatial dataset.

There are many metrics for quantifying daylight availability (Kota and Haberl, 2009). The Split Flux Method calculates the daylight factor, given as a percentage of the total natural light available from an unobstructed overcast sky that reaches points on a work plane within a room. This is a relatively simple metric, but it provides a useful relative measure of the effect of the room design and layout without requiring detailed location-specific weather data.

The main aim in designing for daylight is to maximize daylight values in those areas of the room that require lighting, with the typical tradeoffs against aperture size, construction and layout restrictions. Thus, the key metric is the actual spatial distribution of daylight levels across the workplane. Secondary metrics are the maximum, minimum and average values, as well as the percentage of the total area above or below different values.

To display this information, a two dimensional grid of calculation points is arrayed at a configurable height above the floor to represent the work plane. Values are colour mapped across this grid and contours lines and bands can be superimposed. In addition, a histogram chart displaying the percentage area of the grid within each contour band is displayed immediately adjacent to the main legend, as shown in Figure 5. It is also possible to switch this to a cumulative distribution chart.

4.2.1 Daylight design target

In some cases, a daylight design requirement involves the specification of a minimum percentage of the work plane area that must be at or above a given daylight factor. This information can be derived from both the histogram and the cumulative distribution, but it is more effective if it can be directly visualised within the model.

To do this, a daylight design goal can be set which causes a percentage area marker to be displayed within the histogram chart and the exact area of the work plane at or above the target value shown as a contour within the context of the model. Figure 5 shows an example of this. The target value can be interactively manipulated using either the numeric input block in the popover or by dragging the percentage area marker in the histogram chart.

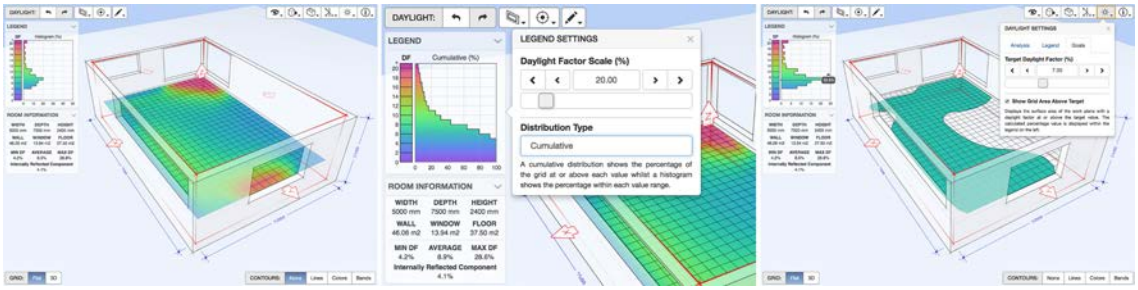

Figure 5: The display of dynamic statistical information about the daylight distribution and, on the right, showing the percentage of the work plane area at or above a target daylight value.

6. Next steps

The first phase of this research is now well underway with most of the core framework in place and functional implementations in Java, JavaScript, C# and C++ to support multiple games engines. Generators for a range of charts and graphs have been developed that can be fed data from the output files of several different analysis tools, as well as a set of interactive manipulators and user interface elements for controlling and interacting with them. Now that the solar geometry and spatial analysis components have been developed, the next set of educational tools will explore the visualisation and quantification of temporal overshadowing and solar availability.

One of the most challenging parts in the development of the next set of tools has been obtaining user-generated building geometry in a form suitable for visualisation and technical illustration styles that require both surface facet and outline edge information. Whilst it is relatively simple to obtain detailed triangulated geometry from a CAD or BIM model, accurately determining the boundary outline of each planar facet is often much more difficult.

An initial solution has been to develop a native export plugin for Revit and code for parsing gbXML files as these allow the one process to extract triangulated mesh as well as surface facets and edge data. The next step is to include Industry Foundation Classes (IFC), which are currently handled by storing them in a BIMserver (BIMserver.org, 2016) database and then using a Render Engine plugin to produce the required geometric data. These three sources are critical as they also allow for the extraction of analytical models with more detailed spatial, zonal and material data which can be used to drive many of the external building analysis tools that generate building performance data.

In addition to the dynamic interactivity that games engines allow, direct access to the model rendering process has been the most compelling feature of this technology. The ability to write custom shaders has already allowed this research to generate a range of analysis, visualisations and non-photorealistic technical illustration techniques that would not have been possible without it.

This now forms the basis for the next phase in which these visualisations and techniques are objectively assessed.

References

Beautypi (2016) ShaderToy, Retrieved from < https://www.shadertoy.com/browse/>.

Bederson B.B. and A. Boltman, A. (1999) Does animation help users build mental maps of spatial information? in *Proceedings of IEEE Symposium on Information Visualisation (InfoVis '99)*, 28– 35

BIMserver.org (2016) Open source BIMserver | in the heart of your BIM, Retrieved from <http://bimserver.org/>.

Chou, Y. (2015) *Actionable Gamification - Beyond Points, Badges, and Leaderboards*, Octalysis Media, ISBN-13: 978-1511744041.

Gonzales. C. (1996) Does Animation in User Interfaces Improve Decision Making? In *Proceedings ACM CHI 1996*, Vancouver, 27-34.

Fagerholt E. and Lorentzon M. (2009) Beyond the HUD -- User Interfaces for Increased Player Immersion in FPS Games, Masters of Science Thesis, Department of Computer Science and Engineering, Chalmers University Of Technology, Göteborg.

Github Inc. (2016) Javascript library for general purpose computing on GPU, Retrieved from <https://github.com/stormcolor/webclgl>

Heer, J. and Robertson, G. (2007) Animated Transitions in Statistical Data Graphics *Proceedings of IEEE Symposium on Information Visualisation (InfoVis '07)*.

Hopkinson, R.G., Petherbridge, P. and Longmore, J. (1966) *Daylighting*. Heinemann, London, UK.

Kota, S. and Haberl, J. S. (2009) Historical Survey of Daylighting Calculations Methods and Their Use in Energy Performance Simulations, Proceedings of the Ninth International Conference for Enhanced Building Operations, Austin, Texas

Lowe, R. and Schnotz, W. (2008), *Learning with Animation: Research Implications for Design*, Cambridge University Press, UK.

Marsh, A.J. (2003) Computer-Optimised Shading Design, *Building Simulation 2003, Eighth International IBPSA Conference*, Technische Universiteit Eindhoven, Eindhoven, Netherlands.