Let There be Light! Knowledge-Based 3-D Sketching Design Tools

Ellen Yi-Luen Do and Mark D. Gross

Let There be Light! Knowledge-Based 3-D Sketching Design Tools

Ellen Yi-Luen Do and Mark D. Gross

This paper presents a framework for 3D knowledge-based sketching tools for lighting design and two software prototypes built to illustrate sketch-based interaction with intelligent systems in 3-D domains. Spot supports direct sunlight simulation and visualization in a selected time period and Light Pen supports placement of electric lighting designs to light an intended area in space. In both examples, a 3-D sketching front-end is coupled with a back-end knowledge-based system. This enables a designer to pose a problem by drawing onto a 3-D model to which the knowledge-based system offers a solution – in one case by providing quantitative data analysis; in the other by modifying the 3-D model. Spot and Light Pen's specific domain of architectural lighting design exemplifies a more general class of 3-D interaction with intelligent systems.

1. Introduction

1.1. Motivation – tools for lighting design

Lighting design is an important part of architectural design. Architects consider light as the "fundamental basis of architecture" (Le Corbusier): "we only know the world as it is evoked by light..." (Louis Kahn); light is part of the "structure of the thinking of the architecture" (Richard Meier), and the wall is a "luminous vertical surface" (Carlo Scarpa). Light is a design variable that affects the character of life and activities in the spaces people inhabit. The orientation, form, and scale of a building, the arrangement of openings and glazing as well as spatial configurations of rooms are strongly influenced by architects' approaches to light.

Lighting design requires the consideration of human needs such as comfort and aesthetics, and the solutions for energy efficiency and cost effectiveness. The balancing act of the quantitative and qualitative concerns of lighting design remains a difficult challenge. Architects have traditionally dealt with lighting design in buildings qualitatively and intuitively. Recent advancements in computer graphics provide compelling rendering tools for photo-realistic visualization to support qualitative assessment. Research and software applications for numerical calculation of diurnal and annual cycles for sunlight now provide methods for quantitative analysis and assessment. Both are welcome developments. However, the question is not whether quantitative or qualitative methods provide better support for lighting design. Designers need to approach lighting both qualitatively and quantitatively. A good design tool should provide relevant information, be integrated into the design environment to better support quick iterative explorations essential to a design decision-making process. To address this need, we propose a framework of a computer-aided 3D sketching environment with knowledge-based lighting design systems.

This paper presents two software prototypes developed under the framework that link intuitive user interfaces to algorithmic calculations that operate on a 3D model. To address the essential pair – the quantitative and the qualitative – we have chosen to implement software modules to separately investigate daylight simulation and lighting fixture design, using sketching in three-dimensional space to manage the two modules. Spot renders daylight analysis and visualization over time on any surface in space. Light Pen recommends lighting fixture placements from intended illuminated surfaces. Below we briefly provide our rationale for building these computational tools for lighting visualization, analysis and design.

1.2. Daylight analysis

Visualization and analysis of sunlight effects in buildings is essential for daylight design and performance. Sunlight has been considered an important feature for any successful space design as well as an energy source for buildings [1]. Louis Kahn stressed the importance of sunlight and argued that "A room is not a room without natural light". Daylight design involves considerations of the variation of lighting conditions and the degree of penetration into buildings. Researchers have developed many methods to help quantitatively assess projected lighting levels. Hand calculation methods such as Lumen methods, daylight protractors, and work sheets are useful tools for calculating physical parameters such as the size and shape of a skylight. Physical model and tilt tables also help analyze full load lighting hours and density. However, applying any of the methods above is labor intensive and involves several steps.

Recently we have seen ray tracing and other computer graphics techniques being employed (e.g., Radiance) to simulate the behavior of light in the real environment. These tools can simulate lighting effects of a space at a given moment in a viewpoint. However, they do not provide real time feedback of lighting effects over time. A high-resolution image of output from a simulation requires complex computation to decide the luminance value and usually takes time to generate. During the early conceptual design stage designers need a simple-to-use simulation and visualization tool to help them understand and recognize lighting problems and opportunities. Rapid direct sunlight simulation in the 3D design environment would be useful in early stage design. Therefore, we implemented the Spot system to enable architects to sketch in 3D to specify areas for quick previews and calculations of the amount of direct sunlight projected over time.

1.3. Lighting fixture design

A lighting designer's main task is to select lighting fixtures and position them in the building appropriately for the activities that are to take place. The location, size, and orientation of windows and skylights must be considered, but typically the architect has already made these building design decisions. Typically, a lighting design includes three categories of illumination: ambient, task, and accent lighting. Ambient illumination supports general activities in space, such as wayfinding, orientation, and movement. Task illumination supports specific activities such as reading, work, eating, and conversation. Accent lighting highlights points of interest, such as paintings, photographs, or architectural details. Several alternatives exist for implementing each type of lighting, for example recessed ceiling lights, fluorescent lamps indirectly reflecting, pendant lamps, floor and table lamps, and track lights. Each has specific preconditions for installation, advantages, and drawbacks. The lighting designer must also specify the configuration and location of switches [2].

1.4. Structure of the paper

In the following sections, we briefly introduce Space Pen, a platform that we built earlier for browsing and sketch annotation in 3-D and explain how it led to Spot and Light Pen. Next we demonstrate Spot and Light Pen and outline the lighting design expertise that these programs embody. We close with a brief discussion about how 3D sketching interface enables easier interaction in the specific domain of architectural lighting and directions for future work.

2. Sketching as an interface to lighting design tools

2.1. Intelligent sketching systems

Generating 3-D geometry from 2-D sketches has been a 'holy grail' research topic for some time, yielding diverse approaches to pen-based interfaces for generating and editing three-dimensional models. Sketch! [3], like its commercial cousin, Sketchup, recognizes gestures and generates modeling commands rather than parsing and interpreting a line drawing, whereas Teddy [4] uses a simple heuristic to inflate three dimensional curvilinear forms from a freehand drawing. VR Sketchpad [5] combines extrusion and symbol recognition to rapidly create VRML worlds. Chateau [6] tries to anticipate the user's intentions in generating a three-dimensional model, offering alternative 3-D completions as the user draws a 2-D sketch. Digital Clay [7] uses constraint propagation of concave and convex vertices to generate three-dimensional models from a two-dimensional diagram. Stilton [8] enables designers to draw into a three-dimensional scene represented in VRML; the program parses the line drawing on the fly to add 3-D geometry to the scene.

Beyond sketch-to-3-D geometry creation, a quite different line of sketching research explores *using diagrams to interact with applications* such as simulation programs, databases and other intelligent systems. This approach is especially appropriate in domains where drawings are common representations. Diagrammatic interaction has been explored in physical domains such as mechanical engineering [9] and geography [10] as well as to support tasks such as interaction design [11] and military action planning [12]. In this vein our earlier work developed a general system for diagrammatic interaction, based on an end-user programmable visual language [13].

Diagram interfaces for intelligent systems have mostly supported two-dimensional drawing. However, in physical design domains such as mechanical engineering, entertainment, and architecture the artifacts that designers manipulate are typically 3-D computer graphics models. Designers operate on these models with applications that process 3-D data and display results in 3-D. For example, a mechanical engineer may perform kinematics analyses of a mechanism in 3-D or finite element analysis of a structure; a game designer may perform visual analysis of sight-lines; and an architect may test a proposed building for emergency egress routes. These analysis tasks typically involve running a 3-D model through a stand-alone application.

We would like to bring applications that operate on 3-D models into the designer's working environment to seamlessly integrate design and analysis. Designers often begin work with informal sketches and diagrams. That is why we want to employ pen-based interfaces to interact with the intelligent systems that serve as computational assistants in three-dimensional design domains. We built Spot and Light Pen to demonstrate how such an interface can be used in the domain of architectural lighting design.

2.2. Space Pen

Our platform for 3-D sketching is Space Pen [14] software we built to support Web based design collaboration with annotation capabilities in 3-D. The Space Pen server converts any VRML model posted by the architects into a Java 3D model in a standard Web browser. Collaborating team members can then browse and annotate by drawing on model surfaces. For example, a team member reviewing a proposed architectural design draws on a wall-graffiti style-to indicate a proposed location for a new window. Space Pen also supports text annotation, with threaded discussions linked to Post-It(r) style tags left in the model. Designers mark on existing model surfaces or on a temporary drawing plane to add geometry to the model. Space Pen identifies figures such as arrows, rectangles, and circles, which it can then rectify as model geometry or interpret as commands. In short, the Space Pen provides a platform for drawing onto and into 3-D models.

2.3. Sketching with light

Our work on Light Pen began after we demonstrated Space Pen [15] to a professional lighting designer [16]. We wondered how to improve Space Pen's rendering capabilities to support lighting design tasks. She commented that commercial lighting design software applications render excellently the visual appearance of a proposed lighting design configuration. Given positions of a set of lighting fixtures, windows, and architectural geometry, the software renders the resulting lighting effects at a given date and time. What lighting designers *really* do, she explained, is "paint with light"-they identify desired lighting effects at specific locations, then *reason backward* to determine the selection and location of lighting fixtures that produce these effects. We realized that the Space Pen could be an effective interface for lighting designers to specify design intentions.

Inverse calculation of lighting effects, working backward from lighting effect to position and characteristics of lighting sources, is not new. The appositely named Painting with Light system [17] computes color and intensity values for fixed theater lighting based on a lighting effect that the scene designer paints on a model of the stage surface. Inverse calculation is also used in other design domains. For example, mechanical engineers use inverse kinematics to determine the geometry of a mechanism needed to produce a given set of motions.

This idea-sketching on a 3-D model to identify desired lighting effects-sparked the development of the Light Pen. More generally, we saw that sketching in 3-D could be a direct and natural means to interact with systems that reason about and calculate on three-dimensional models. We proposed to use light-painting as input to an automated design assistant that would help an architect select and position lamps to produce desired lighting effects.

2.4. Sketching for daylight visualization

Our work on Spot began with the interest of providing sunlight visualization to 3D space and to complement Light Pen's lighting fixture design features. Spot follows the same framework, using Space Pen's Java 3D sketching, navigation and annotation platform as an interface to knowledge based systems. Spot provides direct sunlight visualization in a navigable 3D space. It computes 3D geometry (spatial variables: X, Y, and Z) and also the sun angle variations with the diurnal and annual cycles (temporal variables: date and time).

Multi-dimensional data is usually displayed on a single 2D visualization pane. The Space Series project [18] uses a focus-plus-context technique to support display of spatial and temporal data variations. For lighting experts with specialized knowledge this 2D display is sufficient. However, easy visualization and interaction techniques would better support architects' qualitative assessment when designing in 3D. Software like Ecotect [19] displays the pattern of multiple shadows projected during a period of time on a single diagram, but it does not quantify the amount of received light.

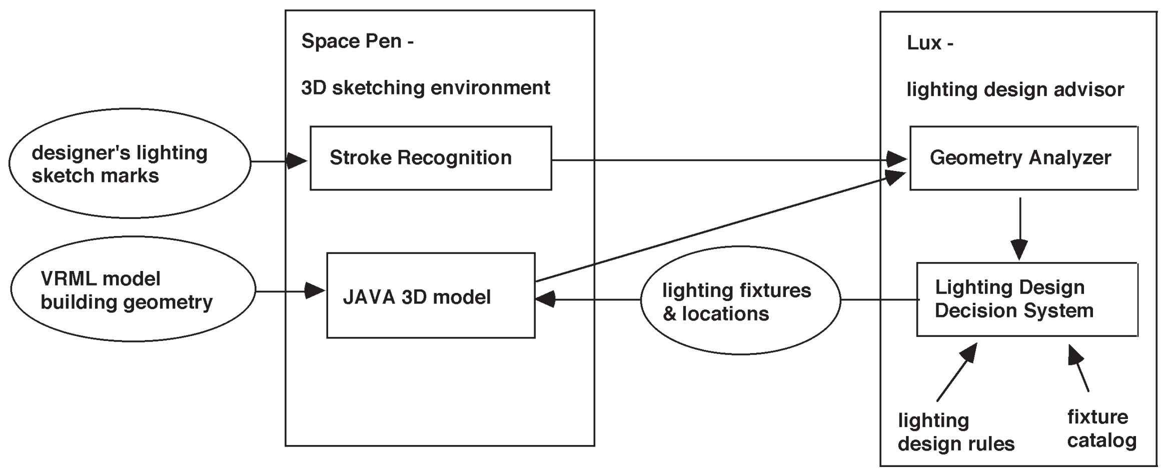
To initiate lighting visualization in Spot, designers first sketch a boundary shape on the 3D model indicating the area for simulation. Spot then generates a representation of the spatial distribution of the illumination level on the selected surfaces over time. Spot also enables designers to visualize the light distribution over time for a given point. For each point tapped by pen (or clicked by mouse) on the 3D model, Spot generates a calendar chart where the X and Y axis represent the months of the year and the time of the day. The color of each calendar cell corresponds to the calculated amounts of light reaching the point the user tapped.

3. Light Pen

The Light Pen system consists of a 3D sketching front end to a rule-based electrical lighting fixture advisor. Below we briefly describe the system architecture and the components of Light Pen, and a use scenario to demonstrate how it supports lighting design. More technical detail on implementation can be found in (Jung, Gross and Do, 2003). Here we provide a conceptual explanation of the work and the current extensions of the rules.

3.1. System architecture

Figure 1 illustrates Light Pen's system architecture. It consists of two communicating components, the Space Pen 3-D drawing program (left), and the Lux lighting advisor (right).



▲ Figure 1. Light Pen's two components: a 3-D sketch browser (Space Pen) and a knowledge-based advisor (Lux)

The designer interacts directly with Space Pen, which provides tools for 3-D browsing and sketching. After importing a three-dimensional (VRML) model the designer marks it up to indicate desired lighting effects. The model geometry and the designer's lighting sketch marks are passed to the Lux lighting design advisor.

Lux is Light Pen's 'back end' intelligent system, coded in Java as a simple set of lighting design decision rules. Lux accepts the lighting sketch marks and building geometry as inputs. Based on the desired lighting and the model geometry Lux recommends solutions, selecting fixtures based on their desired characteristics. Finally it passes these recommendations back to Space Pen, which adds the fixtures to the 3-D model to indicate Lux's proposed design solution.

3.2. The Lux lighting design advisor

Lux is the 'back end' intelligent system component of Light Pen. Lux first analyzes the model geometry on and near the designer's lighting sketch-mark. It determines, for example, whether the area to be lit is a floor, wall, or work-surface, and whether the area is large or small. Next, based on this analysis, Lux determines whether the lighting category is task, accent, or ambient. Then, the system selects a set of appropriate lighting fixtures, based on the lighting category and the architectural geometry. For example, it will suggest track-lighting only if it finds an appropriate surface nearby for mounting the lighting track. Finally, for each fixture that it deems appropriate for the lighting category, Lux identifies an appropriate location. These steps are described in the following sections.

Analyze the model geometry

The architectural model is imported from a CAD program in VRML format so it consists merely of a set of surfaces with no information about what building components or furniture the surfaces may represent. Rather than trying to classify all surfaces in the model *a posteriori*, or requiring the designer to tag them *a priori*, we chose to perform a local analysis on the parts of the model where the designer has sketched lighting marks. The Lux rules use simple predicates to reason about the local architectural geometry of the illumination problem. These predicates include tests for coplanarity, vertical and horizontal surfaces as well as more specific tests to determine whether a surface is a floor, a work-height surface, or near the ceiling.

The designer's marks on the 3-D model are passed as a query to the Lux lighting design advisor, which takes three actions:

- First, it *identifies the type of illumination needed*, based on the size of the light mark and the surface it is drawn on.
- Second, it *selects a lighting fixture* suitable for the illumination type; e.g., track lights for task lighting; a pendant lamp for ambient illumination.
- Third, it *proposes appropriate positions* for mounting the lighting fixture.

This information is conveyed back to the Space Pen, which adds the proposed design elements in their positions to the model and displays the new scene, indicating with a cone of light the illumination effects they provide.

Identify illumination task

Based on the analysis of architectural geometry and the size and location of the lighting sketch mark, Lux then determines the desired category of illumination. It selects "task lighting" if the surfaces to be illuminated are horizontal, and the surface is close to the ceiling. Otherwise, if the area to be lit is small it selects "accent lighting". If the area to be lit is large or the surfaces to be lit are both horizontal and vertical, then Lux selects "ambient" as the illumination category.

Select Appropriate Fixtur

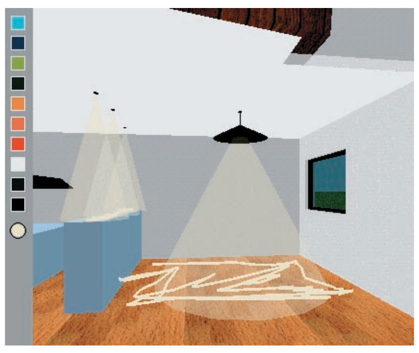
Next, using a decision tree, Lux selects a fixture or a set of fixtures appropriate to the illumination category that it determined in the previous step. If the illumination category is "task lighting" then the set of possible fixtures includes several possibilities: [spotlight, desk lamp, table lamp, fluorescent light, long pendant light]. If the illumination category is "accent" then the only fixture it suggests is "spotlight". If the illumination category is “ambient” then the set of fixtures includes [pendant light, floor lamp, spotlight, fluorescent light]. Lux considers the local architectural geometry in selecting appropriate fixtures from these sets.

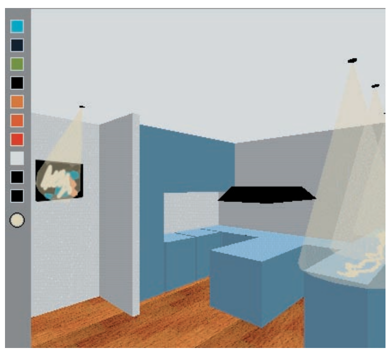
Place fixture in model

Finally the Lux lighting design advisor positions the chosen fixture into the model. The previous fixture selection step guarantees that an appropriate surface exists for mounting the fixture. Still, Lux must propose an exact position so that the fixture can be added to the 3-D model. The choose-fixture-position method takes as arguments the surface to be illuminated, the lighting sketch mark, and the fixture type. A vector is drawn from the surface to be illuminated starting at the center of the lighting sketch mark. The first surface that the vector intersects will be the mounting surface for the fixture and the intersection point will mark the location of the fixture. For a track light, a line along the long direction of the lighting sketch mark generates a corresponding position on the mounting surface.

3.3. Light Pen at work

Figure 2 shows the Light Pen system in use. Using Space Pen, the designer has posed a design problem by ‘sketching light’ on surfaces in the model where lighting is desired: on the floor in the middle of the room and on the kitchen counter.


▲ Figure 2. Designer sketches light; Light Pen recommends solutions


▲ Figure 3. Requesting illumination for a picture

In response, the system proposes lighting fixtures and locations. The Lux lighting advisor recognizes that ambient lighting is needed and proposes a pendant lamp hanging in the middle of the room. It proposes track lights to provide task lighting over kitchen work surfaces.

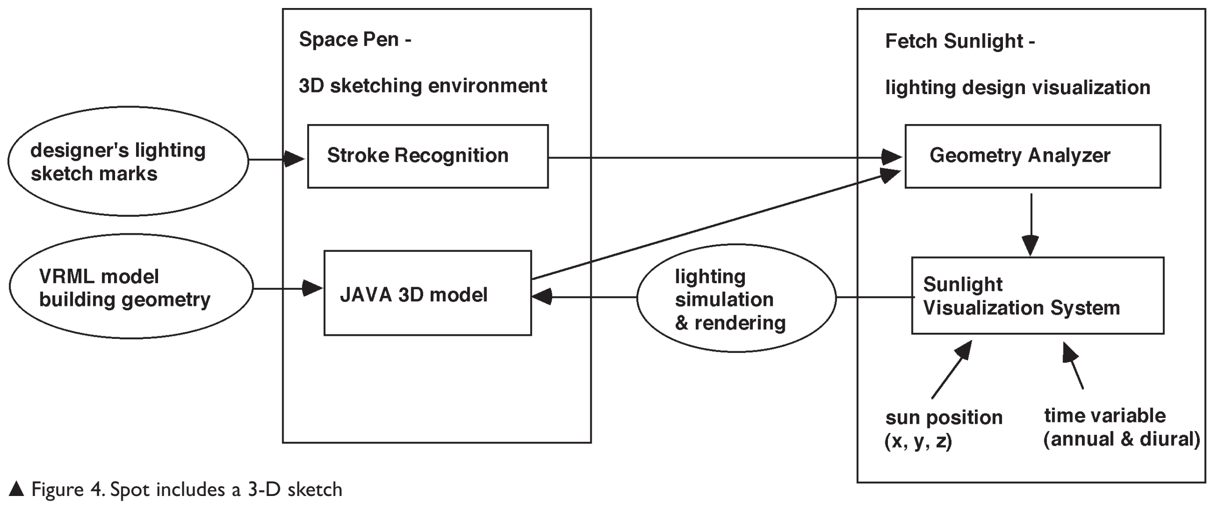
In Figure 3 the designer has moved to a different position in the model and sketched light to illuminate the picture on the wall. Lux suggests a recessed ceiling light fixture.

4. Spot: Fetch sunlight!

Spot is a system that consists of the 3D sketching front end to a rule-based daylight simulation. Below we describe the system architecture and the simulation components. A more complete description of the Spot project can be found in [20]

4.1. Spot system architecture

The implementation of Spot contains two distinct and complementary components: 1) Time Projection and 2) Navigable Animation.


Space Pen -
3D sketching environment
Fetch Sunlight -
lighting design visualization
designer's lighting sketch marks
Stroke Recognition
Geometry Analyzer
VRML model building geometry
JAVA 3D model
lighting simulation & rendering
Sunlight Visualization System
sun position (x, y, z)
time variable (annual & diural)

▲ Figure 4. Spot includes a 3-D sketch browser (Space Pen) and a sunlight simulation system (Fetch Sunlight) that includes spatial variables (x,y,z) and temporal variables (annual and diural)

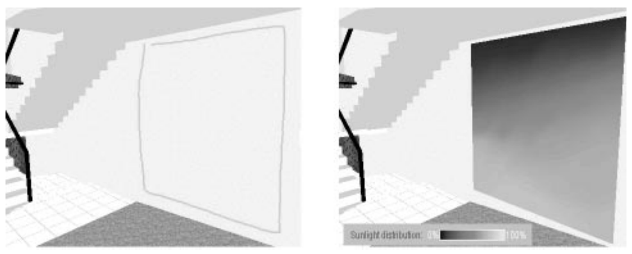
The spatial variables (x, y, z) of 3D geometry are implemented using Space Pen in Java 3D for easy navigation with a standard interface (mouse, arrow keys or joystick) and text annotation and sketching (pen and tablet). The temporal variables (date and time) are displayed in additional views with a look and feel of a 2D graphic calendar. The resulting daylight simulation is displayed on the 3D environment.

4.2. Sunlight distribution in 3D space

Spot supports focused and selective simulation. To specify a surface for simulation, the user draws a boundary area on the 3D model. Spot then paints the selected surface with colors of varied gradients. The colors of the surface's pixels indicate the accumulated cumulative of illumination over time. This interaction is shown in Figure 5. The process of boundary information inference and the cell surface calculation is briefly described below.

Sketch recognition

When a line is sketched on the 3D model, the stroke coordinates are parsed through a shape recognition analysis to determine the closest match (rectangle, circle, or triangle). Once a shape is recognized, it can be rectified as an area for simulation. User can also create a temporary translucent drawing surface by sketching a straight line on any model surface.



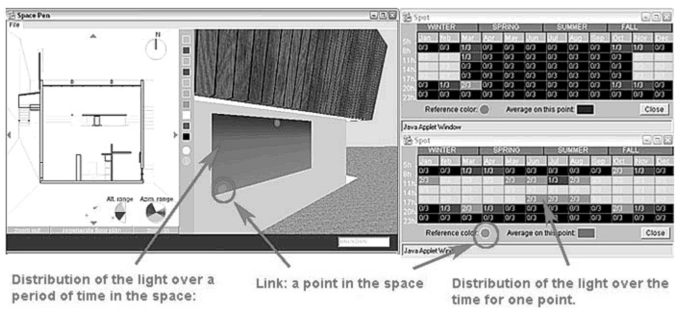
◀ Figure 5. Left: selecting an area for simulation by drawing a boundary shape on the 3D model. Right: shading in rectangle shows illumination result

Defining the surface characteristics for the cells

Once the corners of the stroke's bounding box are known, a loop function implemented in Spot divides the simulated area into cells. The display resolution can be specified in an input window. The simulation processing speed depends on the number of rows and columns. Spot computes illumination and determines the color of each vertex of the cells. The system then interpolates the color of each pixel of the surface. As a result, the surface drawn by Spot is a smooth color gradient. This color display shows the average light intensity for the chosen period of time.

4.3. Time projection

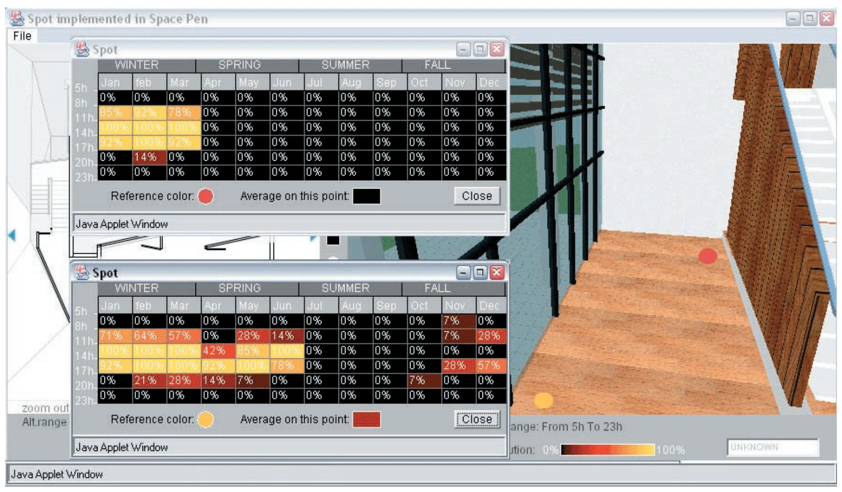
Besides displaying the average illumination values in space, Spot also supports 'behind the scene' data visualization and comparison for any points on the lighting simulation result as shown in Figure 6. Clicking on a point will display a calendar view showing the detail illumination distribution over time. Each cell of the calendar is colored according to the percentage of illumination it receives. Calendars can be generated for any point in the space.


Space Pen
Spot
WINTER
SPRING
SUMMER
FALL
Reference color:
Average on this point:
Close
Java Applet Window
Alt. range
Azim. range
Distribution of the light over a period of time in the space:
Link: a point in the space
Distribution of the light over the time for one point.

◀ Figure 6. The time projection functionality in SPOT: Clicking a point on the simulation result (left) retrieves a calendar view (bottom right) showing the light distribution over annual and diurnal cycles, and average light intensity value. This can be compared with a calendar from another point (top, right)

The pen acts as a magic information wand. When user taps a point on the 3D model, Spot marks it with a sphere with a reference color that serves as index to the corresponding calendar window. The user can click on several points to make a comparative analysis. For example, as shown in

Figure 7, two points of the simulation may result from different light distribution among different seasons.



	WINTER			SPRING			SUMMER			FALL		
	Jan	feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
5h–8h	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
8h–11h	85%	92%	78%	0%	0%	0%	0%	0%	0%	0%	0%	0%
11h–14h	100%	100%	100%	0%	0%	0%	0%	0%	0%	0%	0%	0%
14h–17h	92%	100%	92%	0%	0%	0%	0%	0%	0%	0%	0%	0%
17h–20h	0%	14%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
20h–23h	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

Reference color: 🔴 Average on this point: ■ Close

Java Applet Window

	WINTER			SPRING			SUMMER			FALL		
	Jan	feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
5h–8h	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	7%	0%
8h–11h	71%	64%	57%	0%	28%	14%	0%	0%	0%	0%	7%	28%
11h–14h	100%	100%	100%	42%	85%	100%	0%	0%	0%	0%	0%	0%
14h–17h	92%	100%	100%	92%	100%	78%	0%	0%	0%	0%	28%	57%
17h–20h	0%	21%	28%	14%	7%	0%	0%	0%	0%	7%	0%	0%
20h–23h	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%

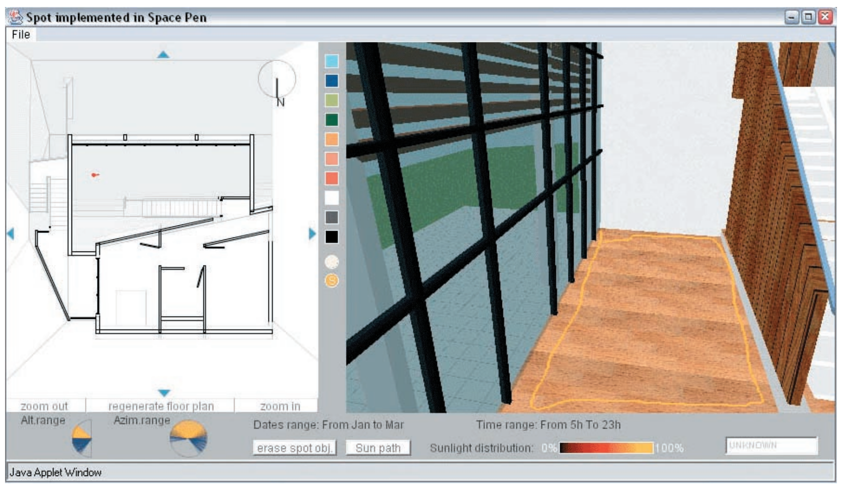
Reference color: 🟡 Average on this point: 🟥 Close

Java Applet Window

▶ Figure 7. Comparing the temporal distribution of light of two given points

4.4. Navigable animation

Spot's Navigable Animation enables the user to interactively visualize shadow casting and animate it through time. Designer can sketch on any surface to indicate the intended area for simulation. For example, Figure 8 (right) shows a rectangle sketch on the floor to indicate the area of interest. A simple ray-tracer implemented in Spot rapidly renders shadows on a selected area.

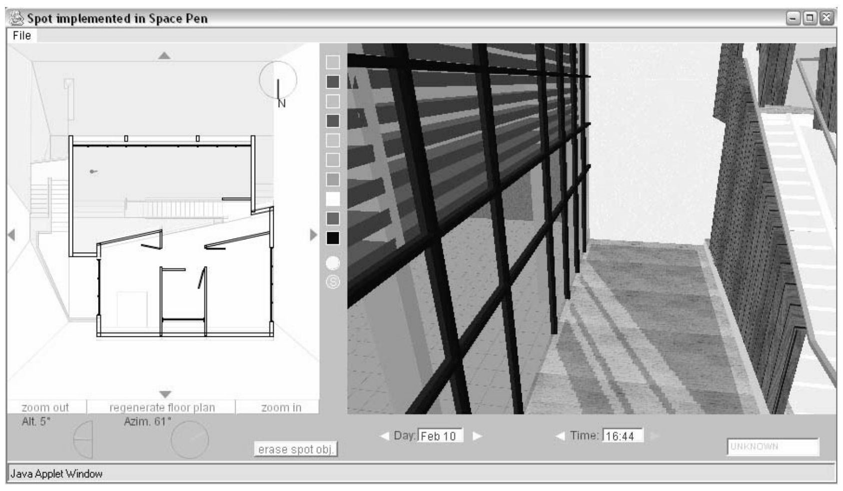

Spot implemented in Space Pen
File
N
zoom out
regenerate floor plan
zoom in
Alt.range
Azim.range
Dates range: From Jan to Mar
Time range: From 5h To 23h
erase spot obj
Sun path
Sunlight distribution: 0% 100%
UNKNOWN
Java Applet Window

▶ Figure 8. Selecting an area for simulation by drawing on the 3D model (right)

4.5. Selective simulation

Instead of waiting for the simulation to render the entire building like many other programs, Spot computes lighting only of the selected area. As a result, Spot renders the shadow casting in real time. Time animation of shadow casting appears just a few seconds after the area for simulation is sketched.

The Java 3D platform provides easy navigation through the 3D environment while viewing the animation in 3D space. Once an area is rendered, the date and time control panel appears on the screen (Figure 9, panel on the bottom). By clicking the forward and backward buttons, designers navigate through the simulation results across date and time. They can examine the shadow effect over time while walking through the 3D space.



◀ Figure 9. Lighting effect animation of shadow casting provides an interface (bottom bar) to move forward or backward through the date (left) and time (right) by clicking on arrow buttons (and moving through 3D space at the same time)

5. Discussion: 3-D interaction with intelligent systems

Our prototypes Light Pen and Spot demonstrate sketching in 3-D as a means to interact with knowledge-based applications (e.g., expert systems, simulations and databases). Both systems use 3D sketching to specify an intended area for analysis or visualization of lighting design. We emphasize that architectural lighting design exemplifies one appropriate domain for this technique, but 3-D sketching has broader application.

We are proposing that sketching to identify illuminated surfaces or area of interest is a natural way to design, and that an interface of 3D sketching that facilitates this approach will be useful. As we move towards pen-based computing that supports interacting with design documents by sketching and 3D visualization, such methods will become increasingly valuable. We believe that 3D sketching could be useful as an interface for knowledge based design systems. For example, we could extend the back end visualization and simulation capabilities of Spot and Light Pen to include thermal and energy analysis, or air ventilation and circulation.

In both Light Pen and Spot sketching is limited to indicating the area where the designer wants illumination or simulation. Sketching serves as an interface to specify the intended focus of attention. We envision, however, that other applications would require an interface that could recognize and interpret a more sophisticated visual language. Currently these systems only employ simple shape recognition (e.g., rectangles, triangles, circles, arrows and lines).

Future work could add recognition and training capabilities of more complex symbols based on configurations of shapes and rule sets. For example, the sketch vocabulary could include symbols for sensors, wall sconces, skylights, fluorescent lights and window shades, etc. Recognition of these symbols could trigger operations to add and modify geometry, or a command to activate simulation or other action. For example, a circle immediately drawn before an arrow may indicate an intention to move (an object) to a new location as specified by a symbol (another circle) drawn immediately after the arrow.

Light Pen, as explained earlier, employs a forward chaining reasoning process to identify the correct surface for mounting the lighting fixture by first recognizing the intended surface for illumination and infer accordingly the shortest path to a wall surface and the angle of incidence. The system advises the design about fixture placement according to guidelines. For example, a task light will be placed directly above the intended illuminated work surface, accent lighting will be projected from ceiling to wall, and ambient light placed to illuminate a larger area of space. Future work could consider typical user locations and reduce glare or reflections for these preferred views. Currently the system can deal with sketching on any single vertical or horizontal plane as well as any two adjacent surfaces of different angles. The system currently places a single light fixture on the ceiling to illuminate both surfaces, however, it would be easy to produce a collection of design alternatives for user selection of preferences as well as linking to a lighting fixture product catalog and specification information.

The system could take into account the designer's sketches of window and skylight openings and lighting fixture placements to generate quick simulation and visualization. Future work could employ simulation engines from commercial software such as Radiance. We chose to implement our own knowledge-based systems instead of using existing simulation systems because they either have a complicated system architecture that requires modification of internal representations to add new functionality or the implementation dictate a certain input format.

Several designers have tested our systems. They found the idea of sketching light into a space is intriguing and argued that it's an improvement over a multi-view 2D representation. They found the usual 2D lighting design reference chart and table useful but distracting for design tasks. The navigable 3D model provides spatial coherence and accessible perspective views that are advantageous over orthogonal projects. In our implementation of the Java 3D navigation we support a game like interface that uses arrow keys to move and pan because some people found the standard VRML navigation difficult. With the Space Pen engine, the viewpoint where one sketches or annotates is automatically saved and displayed on the real-time generated floor plan as an arrow marker. Users of the system or their web collaborators can easily navigate through the space by clicking on any previously saved viewpoints.

While our project does not attempt to create a thorough expert system, it could look at ways to make the information more useful or 'transparent' to the users. For example, it will be an easy extension to provide explanations about how the Light Pen arrived at the specific lighting display suggestions such as a pendant fixture rather a floor lamp. The program could show alternative variations to make the designer aware of new possibilities, rather than defaulting to a certain type of fixture.

Our interest is in coupling a 3-D sketch interface with intelligent systems. Therefore, we built our own calculation and ray-tracing engine for Spot so that we could design and control the interaction as needed instead of being limited by the behavior model of any existing simulation software. The experience in building Light Sketch [21] demonstrated the feasibility, though trivial, to connect with existing software such as Radiance. In Light Pen we built into Lux only a primitive model of lighting design expertise. However, a more comprehensive version would interact with the Space Pen in much the same way. Rather than extend Lux's lighting expertise, we were more concerned with demonstrating this system architecture's general utility. We therefore built a second instance of a 3-D sketch interface, this time to a simulation program. The "Spot" system, also in the architectural lighting domain, enables a designer to pose queries to a daylight simulator by sketching on the surfaces of a building model. Spot responds to these queries by displaying the time-varying lighting effects on the surfaces that the designer has indicated.

These experiments have encouraged us to work toward a general architecture to support 3-D interaction with intelligent systems of various types. Such an architecture might go beyond sketching, and embrace a multi-modal approach including speech and gesture.

Acknowledgements

Funding from the French Region Lorraine (to Sébastien Bund) provided support for our initial ventures into the development of Spot. Light Pen project was supported in part by the National Science Foundation under Grant IIS 00-96138. The views and findings contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We thank the anonymous reviewers and co-editors of this issue (Celine Pinet and Wassim Jabi) from whose helpful comments the paper has benefited.

References

1. Anders, G., *Daylighting: Performance and Design*. 1995, New York: Van Nostrand Reinhold.
2. Egan, M.D. and V.W. Olgyay, *Architectural Lighting*. 2 ed. 2001: McGraw Hill.
3. Zeleznik, R.C., K.P. Herndon, and J.F. Hughes, *Sketch: An Interface for Sketching 3D Scenes*. SIGGRAPH '96, 1996, pp. 163-170.

4. Igarashi, T., S. Matsuoka, and H. Tanaka, Teddy: a sketching interface for 3D freeform design. *Proceedings of the SIGGRAPH 1999 annual conference on Computer graphics*, 1999, pp. 409-416.

5. Do, E.Y.-L., Drawing Marks, Acts and Reacts: toward a computational sketching for architectural design, *AIEDAM – Artificial Intelligence in Engineering Design, Analysis and Manufacturing*, I. Parmee and I. Smith, eds.. 2002, Cambridge University Press: Cambridge, UK, pp. 149-171.

6. Igarashi, T. and J.F. Hughes, *A Suggestive Interface for 3D Drawing, UIST, User Interface Software and Technology*. 2001, ACM, pp. 173-181.

7. Schweikardt, E. and M.D. Gross, Digital Clay: Deriving Digital Models from Freehand Sketches, *Digital Design Studios: Do Computers Make A Difference? ACADIA 98*, T. Seebohm and S.V. Wyk, eds.. 1998, pp. 202-211.

8. Turner, A., D. Chapman, and A. Penn, Sketching Space, in *Computers and Graphics*, 2000, No. 24, pp. 869-876.

9. Stahovich, T.H., R. Davis, and H. Shrobe, Generating Multiple New Designs from a Sketch. *Proceedings of AAAI*, 1996, pp. 1022-1029.

10. Egenhofer, M., Spatial-Query-by-Sketch. *IEEE Symposium on Visual Languages*, 1996, pp. 60-67.

11. Landay, J.A. and B.A. Myers, Sketching Interfaces: Toward More Human Interface Design. *IEEE Computer*, 2001, Vol. 34, No. 3, pp. 56-64.

12. Forbus, K., J. Usher, and V. Chapman, Sketching for military courses of action diagrams. *ACM Intelligent User Interfaces*, 2002, pp. 61-68.

13. Gross, M.D. and E.Y.-L. Do, Drawing on the Back of an Envelope, in *Computers and Graphics, Calligraphy Interface*, J.A. Jorge and E. Glinert, eds., 2000, Pergamon Press: New York. pp. 835-849

14. Jung, T., E.Y.-L. Do, and M.D. Gross, From Redliner to Space Pen. *ACM Intelligent User Interfaces*, 2002, pp. 95-102.

15. Jung, T., M.D. Gross, and E.Y.-L. Do, Space Pen: annotation and sketching on 3D models on the Internet, *CAAD Futures 2001*, B.d.Vries, J.P.v. Leeuwen, and H.H. Achten, eds,. 2001, Kluwer Academic Publishers: Eindhoven, pp. 257-270.

16. Erwine, B., personal communication (April 30), 2002.

17. Schoeneman, C., J. Dorsey, B. Smits, J. Arvo, et al., Painting with Light. *SIGGraph*, 1993, pp. 143-146.

18. Glaser, D. and M. Hearst. Space Series: A focus+context technique for displaying spatial and temporal data. *IEEE Symposium on Information Visualization '99*, Late Breaking Hot Topics. 1999. San Francisco.

19. Roberts, A. and A. Marsh, Ecotect: Environmental Prediction in Architectural Education, in *eCAADe 2001*. 2001, 342-347.

20. Bund, S., A 3D environment for direct sunlight visualization, DEA – "Modelisation et Simulation ees Espaces Batis", Master Thesis, in *Centre de Recherche en Architecture et Ingénierie* (CRAI, Ecole d'Architecture de Nancy, France), conducted at the Design Machine Group (University of Washington, USA). 2003, Ecole d'Architecture de Nancy: Nancy.

21. Glaser, D., J. Voung, L. Xiao, B. Tai, et al., LightSketch: A sketchmodelling program for lighting analysis, *CAAD Futures 2003*, Kluwer, 2003371-382

Ellen Yi-Luen Do and Mark D. Gross, Design Machine Group, University of Washington, 208 Gould, Department of Architecture, Box 355720, Seattle, WA 98195-5720, USA

{ellendo, mdg*@acm.org}