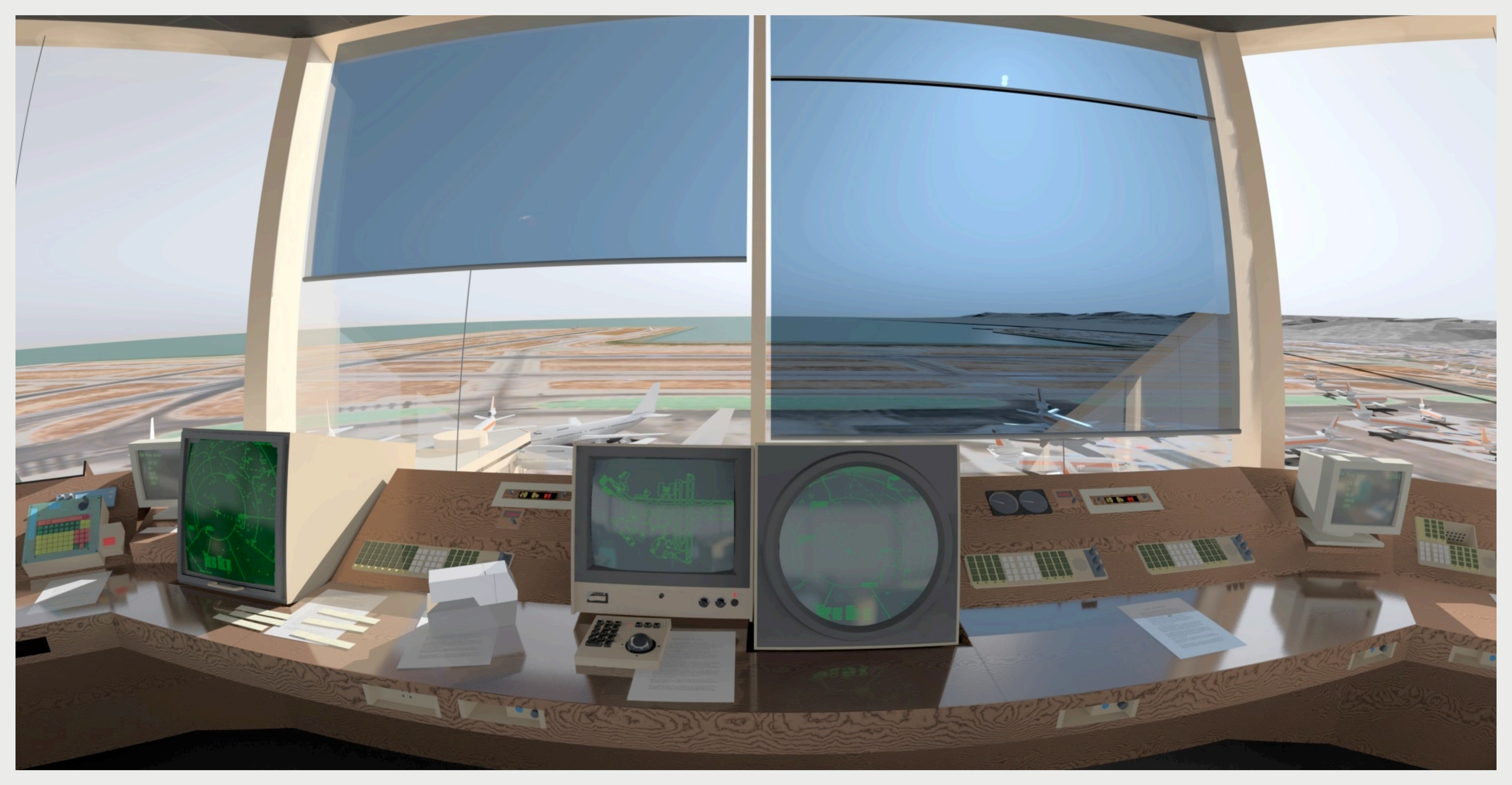
WHAT'S OLD IN RADIANCE?

Greg Ward, Anyhere Software

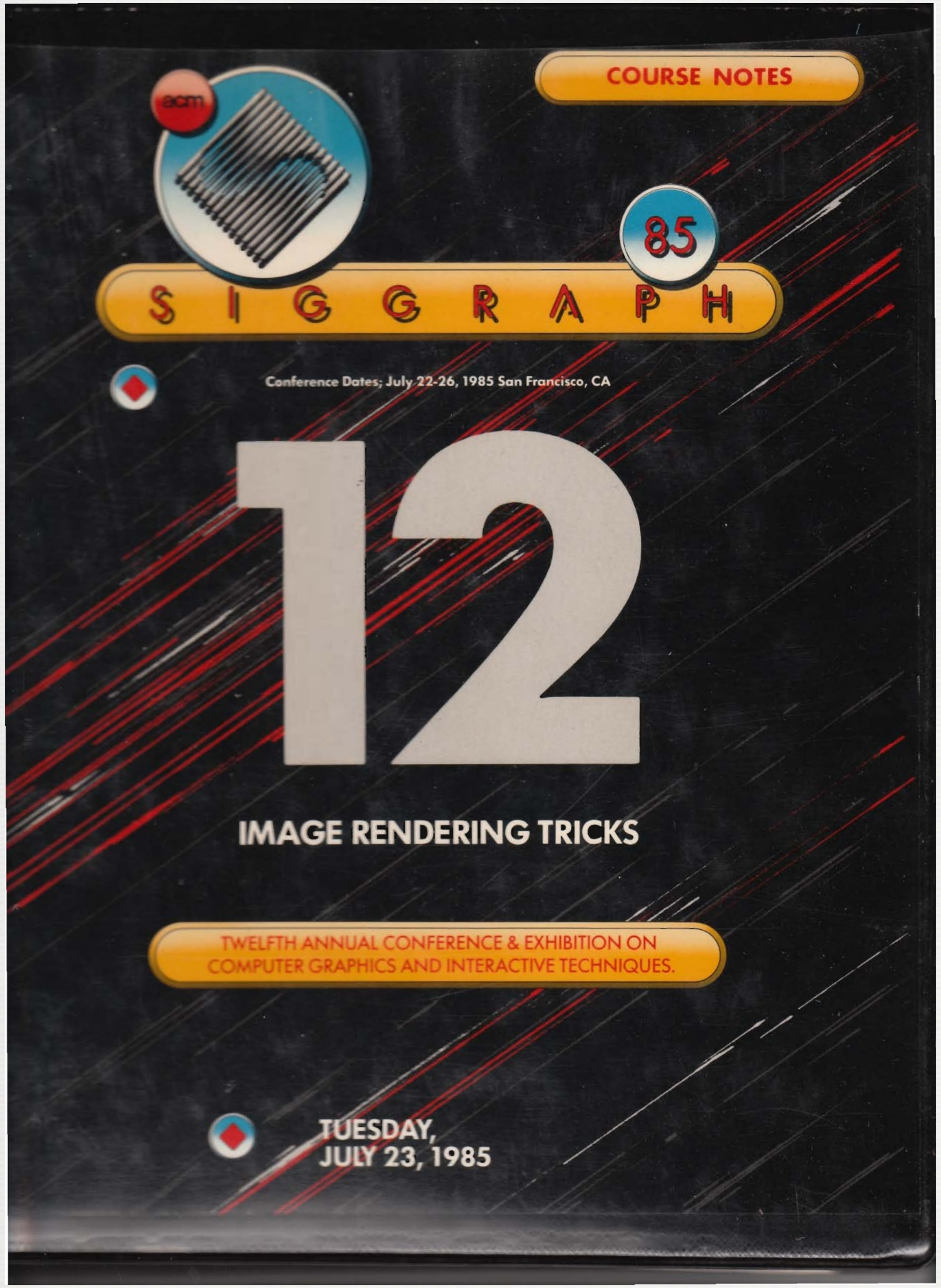


KEYS: INNOVATION, EVOLUTION, & COMMUNITY

- Started from personal interests and curiosity
- Grew with encouragement from LBL Lighting Group
- Developed into a funded DOE software project
- Support & validation from lighting and daylighting community
  - IEA, EPFL, Fraunhofer Institute (Stuttgart & Freiburg)
- Uptake in research, open source, and commercial applications

HUMBLE BEGINNINGS

- *Radiance* grew out of personal interest in computer graphics techniques
  - 1985 SIGGRAPH course, "Image Rendering Tricks"
    - Turner Whitted, Rob Cook, Jim Blinn
- Early experiments and code development "boot-legged" off EPRI project
  - Encouragement from Francis Rubinstein, Sam Berman, Rudy Verderber



Precision, Resolution, and Dynamic Range

Once upon a time graphics programmers at the University of Utah discovered that their PDP-10 executed floating point arithmetic instructions as fast as it did fixed point arithmetic. Rather than worry about overflow and shift counts, they merrily programmed everything using floating point numbers. This probably isn't advisible on most machines.

Dynamic Range, Resolution, etc.

Consider a large spacecraft (roughly 2 kilometers in length). Now imagine rendering this object from a close view in which individual rivets (diameter less than 0.3 cm) are visible. Assume that a convincing rendering of the rivet requires a resolution of 1 part in 10. Then the dynamic range needed to represent this object is (2*10**3)/(3*10**-4) or about 7 orders of magnitude. While this is well within the dynamic range of a single precision floating point number (24 bits of mantissa, 8 bits of exponent), a 32 bit integer would retain more accuracy. A fleet of spacecraft, however, might strain the limits of the long integer. A sixteen bit integer is hopelessly overwhelmed by the dynamic range, although it has perfectly adequate resolution for position in the image plane - 1/64th of a pixel for 1024x1024 resolution.

The key to understanding this example is to note that different representations are needed at different points in the rendering pipeline.

A Guide to Representation

Attribute	Stage	Representation
geometry	before clipping	use floating point for dynamic range
geometry	after clipping	use a short integer for most display resolutions
geometry	scan conversion	tack a 16 bit fraction onto the 16 bit integer to maintain precision
intensity	prior to storage	use signed short integer for each primary color
intensity	stored display	use unsigned byte
intensity	out of video DACS	use 10 bits for compensation

Turner Whitted

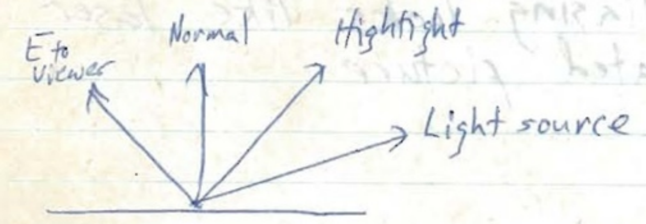
Shaders (Ray Tracing)

Anti-aliasing tricks (only linear systems ?)

Use black if background unknown (for cursor)

Shaders

1) Get intens. value from 1-dim lookup table



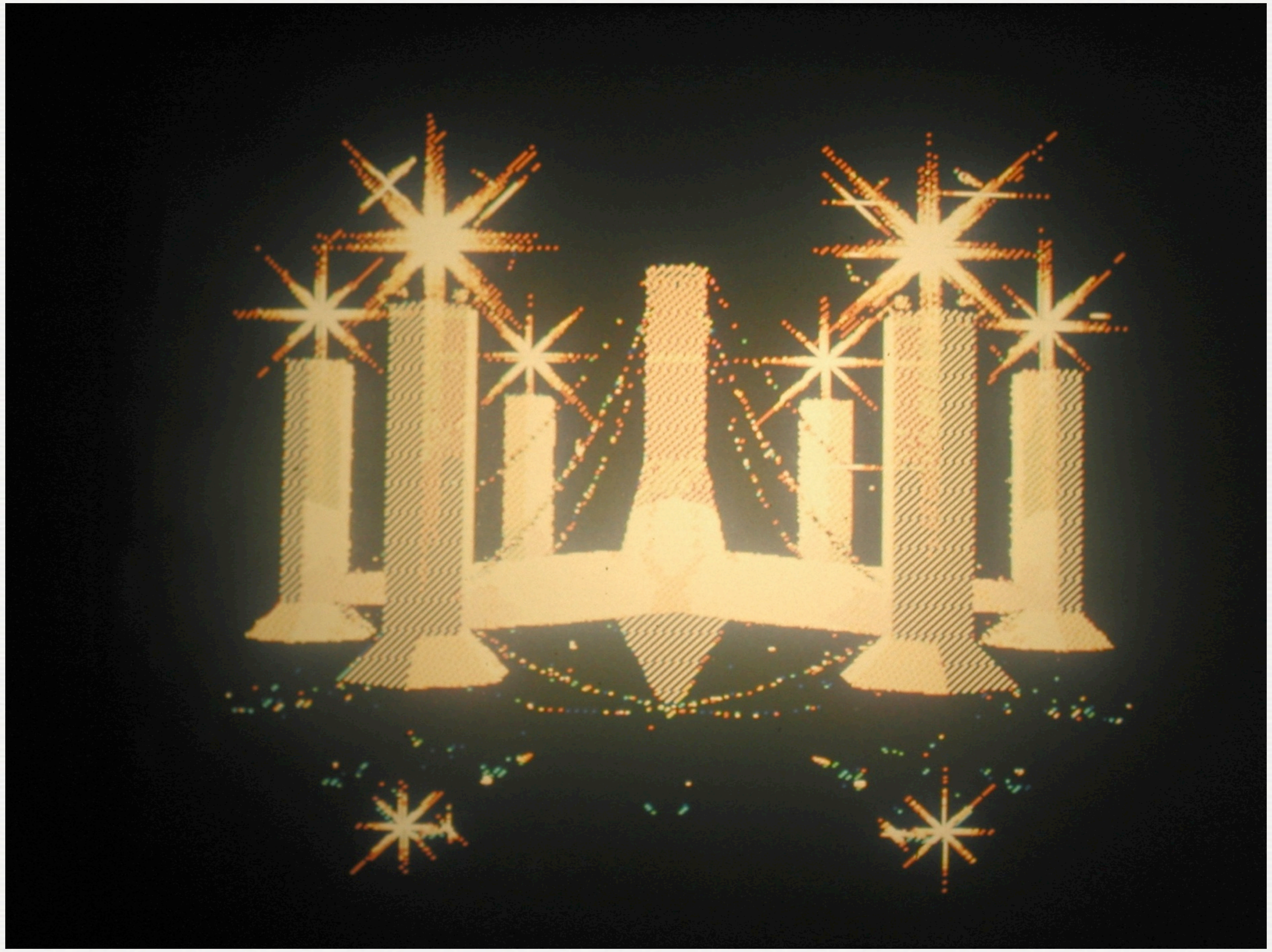
value $= \int_\phi \int_\theta R(\phi,\theta)\, L(x,y,z,\phi,\theta)\, d\phi\, d\theta$

over hemisphere

Reflectance function

Lighting function

FIRST LIGHT (1985-6)



First screen capture (simulated flare)

Simulation of my desk

INDIRECT IRRADIANCE CACHING

- 1986 was “break-through” year, where we found a practical approach to global illumination using ray-tracing (Indirect Irradiance Caching)
  - Pure Monte Carlo had been tried, but it was too slow
  - Caching irradiance integrals made global illumination possible
  - DOE funding followed demonstrated ability in lighting visualization
- 1988 SIGGRAPH paper “A Ray-tracing Solution for Diffuse Interreflection”
- In following decade, IIC became more widely used than radiosity method

1988 SIGGRAPH paper (resubmitted after 1987 rejection)

Computer Graphics, Volume 22, Number 4, August 1988

A Ray Tracing Solution for Diffuse Interreflection

Gregory J. Ward
Francis M. Rubinstein
Robert D. Clear

Lighting Systems Research
Lawrence Berkeley Laboratory
1 Cyclotron Rd., 90-3111
Berkeley, CA 94720
(415) 486-4757

Abstract

An efficient ray tracing method is presented for calculating interreflections between surfaces with both diffuse and specular components. A Monte Carlo technique computes the indirect contributions to illuminance at locations chosen by the rendering process. The indirect illuminance values are averaged over surfaces and used in place of a constant "ambient" term. Illuminance calculations are made only for those areas participating in the selected view, and the results are stored so that subsequent views can reuse common values. The density of the calculation is adjusted to maintain a constant accuracy, permitting less populated portions of the scene to be computed quickly. Successive reflections use proportionally fewer samples, which speeds the process and provides a natural limit to recursion. The technique can also model diffuse transmission and illumination from large area sources, such as the sky.

General Terms: Algorithm, complexity.

Additional Keywords and Phrases: Caching, diffuse, illuminance, interreflection, luminance, Monte Carlo technique, radiosity, ray tracing, rendering, specular.

1. Introduction

The realistic computer rendering of a geometric model requires the faithful simulation of light exchange between surfaces. Ray tracing is a simple and elegant approach that has produced some of the most realistic images to date. The standard ray tracing method follows light backwards from the viewpoint to model reflection and refraction from specular surfaces, as well as direct diffuse illumination and shadows [15]. Accuracy has been improved with better reflection models [4] and stochastic sampling techniques [6]. Unfortunately, the treatment of diffuse interreflection in conventional ray tracers has been limited to a constant "ambient" term. This approximation fails to produce detail in shadows, and precludes the use of ray tracing where indirect lighting is important.

We present a method for modeling indirect contributions to illumination using ray tracing. A diffuse interreflection calculation replaces the ambient term directly, without affecting the formulas or algorithms used for direct and specular components. Efficiency is obtained with an appropriate mix of view-dependent and view-independent techniques.

2. Interreflection in Ray Tracing

Ray tracing computes multiple reflections by recursion (Figure 1). At each level, the calculation proceeds as follows:

1. Intersect the ray with scene geometry.
2. Compute direct contributions from light sources.
3. Compute specular contributions from reflecting surfaces.
4. Compute diffuse contributions from reflecting surfaces.

The complexity of the calculation is closely related to the difficulty of step 1, and the number of times it is executed as determined by the propagation (recursion) of steps 2 through 4. Step 2 requires as many new rays as there are light sources, but the rays do not propagate so there is no growth in the calculation. Step 3 can result in a few propagating rays that lead to geometric growth if unchecked. Methods for efficient specular component computation have been described by [8], [5] and [14]. The diffuse contributions in step 4, however, require many (>100) propagating rays that quickly overwhelm a conventional calculation. Most methods simply avoid this step by substituting a constant ambient term. Our goal is to find an efficient method for computing diffuse interreflection and thereby complete the ray tracing solution. We start with a summary of previous work in this area.

An advanced ray tracing method developed by Kajiya follows a fixed number of paths to approximate global illumination at each pixel [8]. Using hierarchical "importance" sampling to reduce variance, the illumination integral is computed with fewer rays than a naive calculation would require. This brings ray tracing closer to a full solution without compromising its basic properties: separate geometric and lighting models, view-dependence for efficient rendering of specular effects, and pixel-independence for parallel implementations. Unfortunately, the method is not well suited to calculating diffuse interreflection, which still requires hundreds of samples. A high-resolution image simply has too many pixels to compute global illumination separately at each one.

The radiosity method, based on radiative heat transfer, is well suited to calculating diffuse interreflection [12][10][2]. Surfaces are discretized into patches of roughly uniform size, and the energy exchange between patches is computed in a completely view-independent manner. The method makes efficient use of visibility information to compute multiple reflections, and sample points are spaced so that there is sufficient resolution without making the calculation intractable. In areas where illumination changes rapidly, the patches can be adaptively subdivided to maintain accuracy [3]. However, the standard radiosity method models only diffuse surfaces, which limits the realism of its renderings. Immel extended the approach to include non-diffuse environments, adding bidirectional reflectance to the energy equations [7]. Unfortunately, the view-independent solution of specular interreflection between surfaces requires sampling radiated directions over very small (approaching pixel-sized) surface patches. The resulting computation is intractable for all but the simplest scenes.

SIGGRAPH '88, Atlanta, August 1-5, 1988

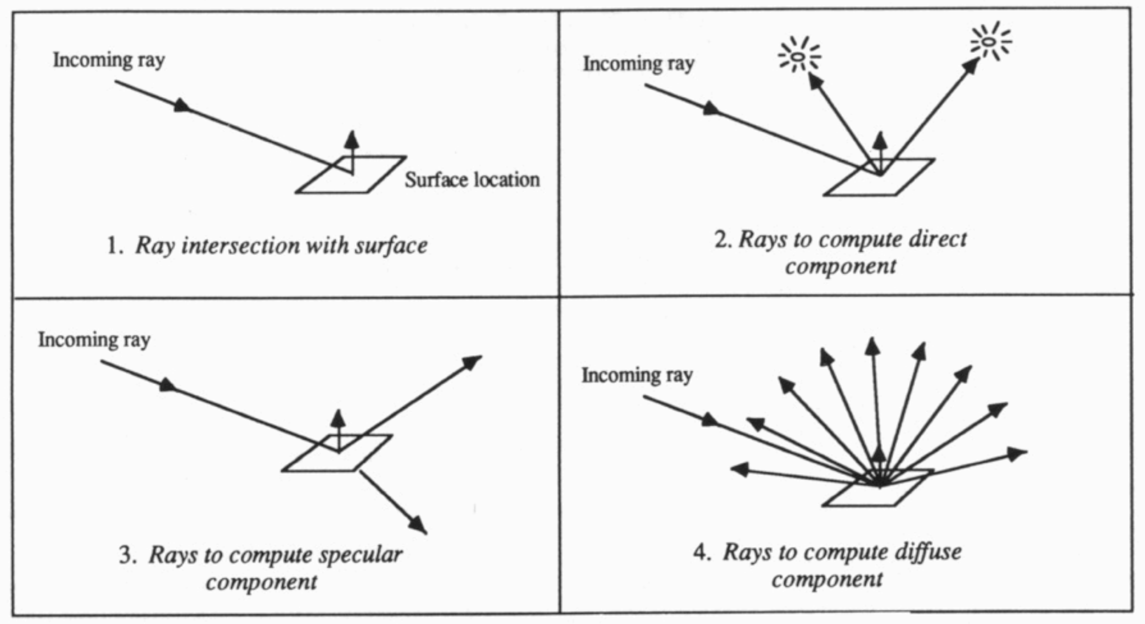


Figure 1: The four steps of ray tracing.

A combined ray tracing and radiosity approach was designed by Wallace to take advantage of the complementary properties of the two techniques [13]. Wallace divides energy transport into four "mechanisms:" diffuse-diffuse, specular-diffuse, diffuse-specular, and specular-specular. He then proceeds to account for most of these interactions with clever combinations of ray tracing and radiosity techniques. Unfortunately, there are really an infinite number of transport mechanisms, such as specular-specular-diffuse, which are neglected by his calculation. The generalization Wallace suggests for his approach is equivalent to *view-independent* ray tracing, which is even more expensive than general radiosity [7].

3. Diffuse Indirect Illumination

Our development of an efficient ray tracing solution to diffuse interreflection is based on the following observations:

- Because reflecting surfaces are widely distributed, the computation of diffuse indirect illumination requires many sample rays.
- The resulting "indirect illuminance" value† is view-independent by the Lambertian assumption [9].
- The indirect illuminance tends to change slowly over a surface because the direct component and its associated shadows have already been accounted for by step 2 of the ray tracing calculation.

For the sake of efficiency, indirect illuminance should not be recalculated at each pixel, but should instead be averaged over surfaces from a small set of computed values. Computing each value might require many samples, but the number of values would not depend on the number of pixels, so high resolution images could be produced efficiently. Also, since illuminance does not depend on view, the values could be reused for many images.

How can we benefit from a view-independent calculation in the inherently view-dependent world of ray tracing? We do not wish to limit or burden the geometric model with illuminance information, as required by the surface discretization of the radiosity method. By the same token, we do not wish to take view-independence too far, calculating illuminance on surfaces that play no part in the desired view. Instead we would like to take our large sample of rays only when and where it is necessary for the accurate computation of an image, storing the result in a separate data structure that puts no constraints on the surface geometry.

In our enhancement of the basic ray tracing technique, indirect illuminance values are cached in the following manner:

```
If one or more values is stored near this point
    Use stored value(s)
Else
    Compute and store new value at this point
```

The computation of a new value uses the "primary method." The technique for finding and using stored values is called the "secondary method." The primary method is invoked to calculate a new value the first time it is needed, which is when the secondary method fails to produce a usable estimate from previous calculations (Figure 2). Determining the appropriate range and presenting a surface-independent storage technique are the two main points of this paper. Before we explore these issues, we present a basic computation of indirect illuminance.

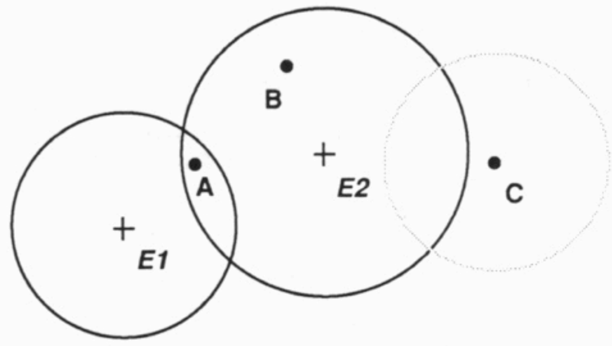


Figure 2: Illuminances *E1* and *E2* were calculated previously using the primary method. Test point **A** uses an average of *E1* and *E2*. Point **B** uses *E2*. Point **C** results in a new indirect illuminance value at that location.

†We define indirect illuminance as the light flux per unit area arriving at a surface location via non-self-luminous surfaces.

Computer Graphics, Volume 22, Number 4, August 1988

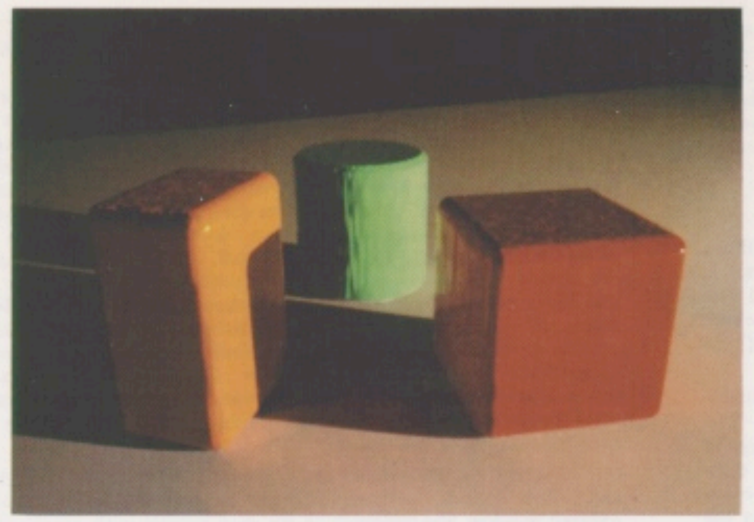


Figure 5a: Colored blocks with diffuse indirect calculation.

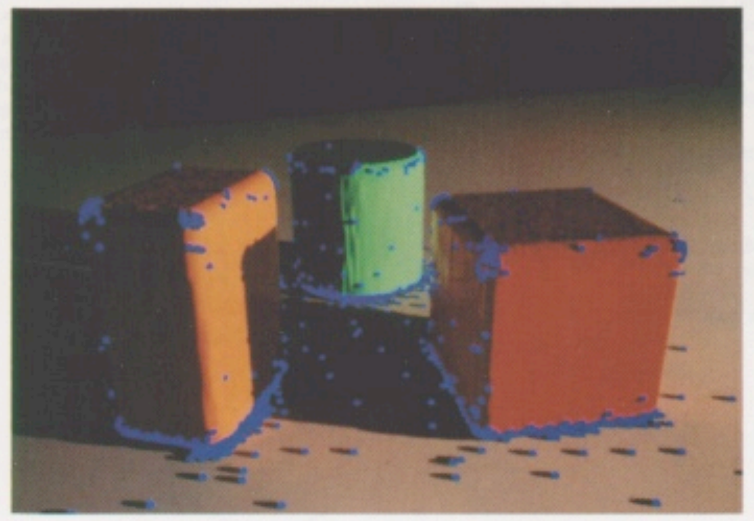


Figure 5b: Blocks with illuminance value locations in blue.

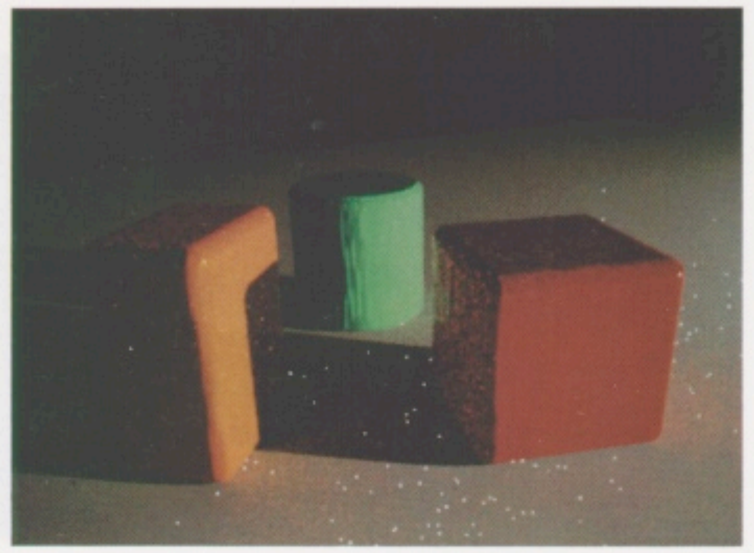


Figure 5c: Blocks using conventional ray tracing techniques.

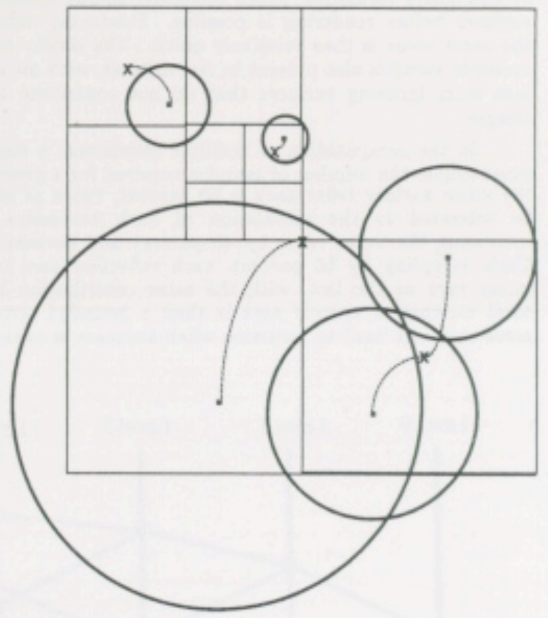


Figure 6: Five indirect illuminance values are shown with their respective domains (circles) linked by dotted lines to the appropriate nodes (squares).

Authors: Greg Ward, Francis Rubinstein, Bob Clear

WHAT'S IN A NAME?

- It helps knowing what to call your software…
- Well-regarded program LUMEN by Dave DiLaura was industry standard
- Similarly, "radiance" is lighting unit corresponding to the value of a ray
  - radiometric units are watts/steradian-meter^2, photometric candelas/m^2

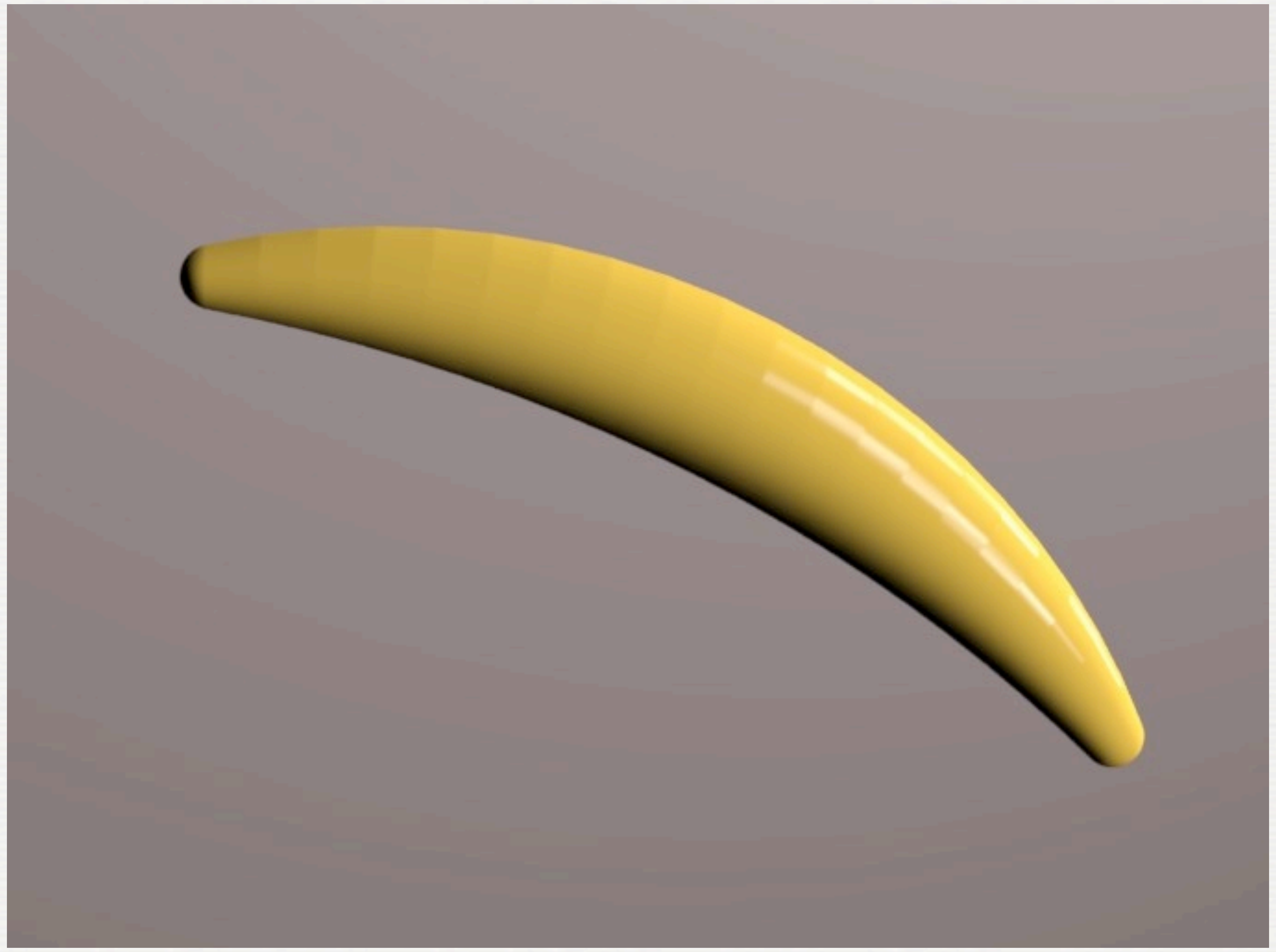
See why I'm not in advertising?

WHAT IS DIFFERENT ABOUT RADIANCE?

- Follows *Unix toolbox* model
  - Small, specialized programs that interact through standard data formats
- Source code applies “least common denominator” approach to portability
  - Allows to grow and maintain cross-platform tools with limited resources
- Took advantage of early adopters with unmet needs and good suggestions

UNIX TOOLBOX MODEL

generate
transform
compile
render
filter
convert



genworm yellow banana '0' '5*sin(t)' '5*cos(t)' '.4-(.5-t)*(.5-t)' 20 | xform -t 70 50 30 | oconv room.rad - | rpict -vf ban.vf -x 2048 -y 2048 | pfilt -1 -x /3 -y /3 -r .6 | ra_tiff - banana.tif

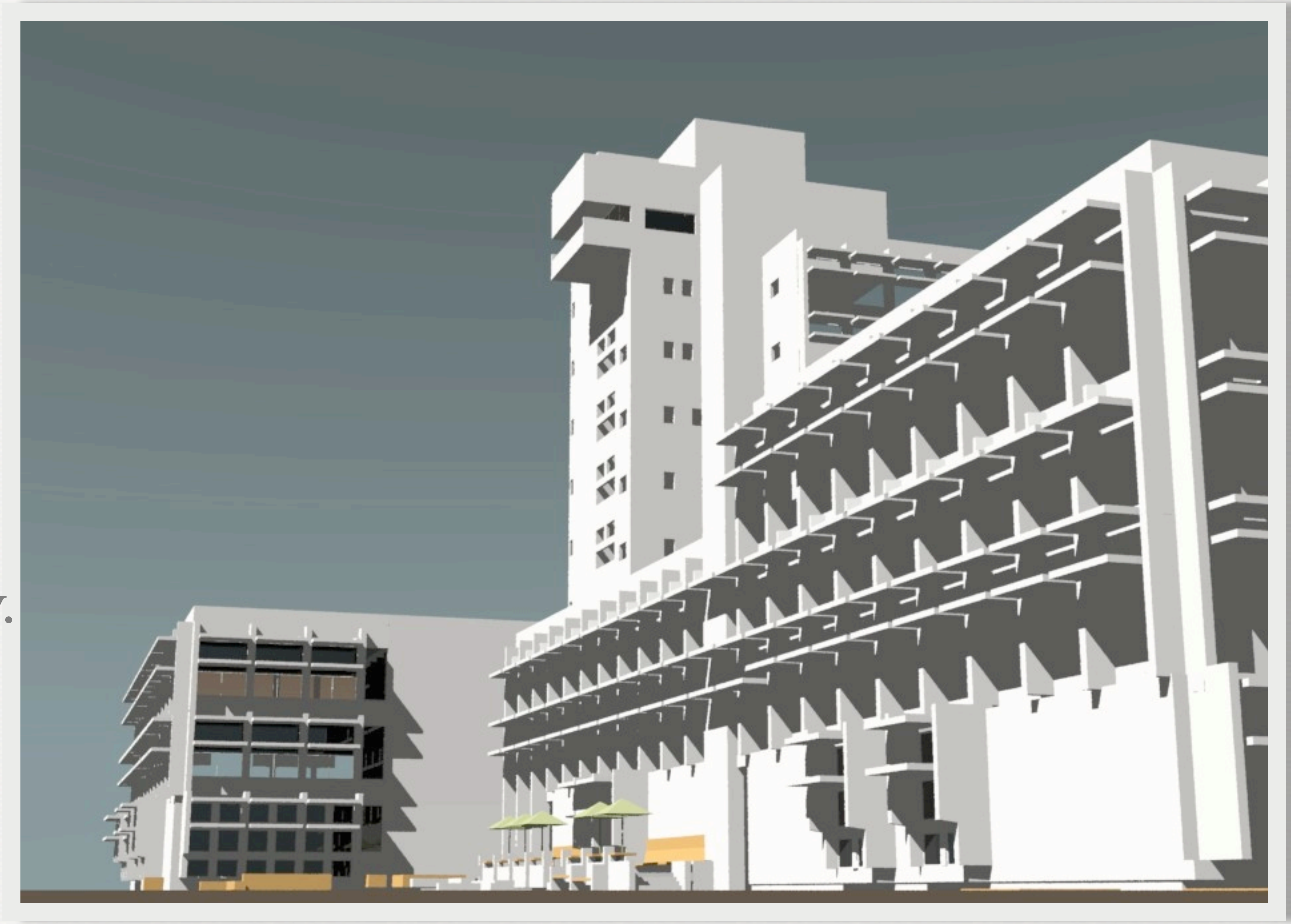
CURRENT LIST OF TOOLS

3ds2mgf
bgraph
bsdf2klems
bsdf2rad
bsdf2rado
bsdf2ttree
bsdfquery
bsdfview
cnt
commake
compamb
cv
dayfact
dctimestep
debugcal
dgraph
dmake
eplus_adduvf
epw2wea
ev
evalglare
falsecolor
fieldcomb
findglare
gcomp
genBSDF
genambpos
genblinds
genbox
genclock
gendaylit
gendaymtx
genklemsamp
genprism
genrev
genrhgrid
gensky
genskyvec
gensurf
genworm
getbbox
getinfo
glare
glarendx
glaze
glrad
histo
icalc
ies2rad
igraph
lampcolor
lookamb
ltpict
ltview
macbethcal
meta2bmp
meta2tga
mgf2inv
mgf2meta
mgf2rad
mgfilt
mkillum
mkpmap
mksource
neaten
nff2rad
normpat
normtiff
obj2mesh
obj2rad
objline
objpict
objview
oconv
optics2rad
pabopto2bsdf
pabopto2rad
pabopto2xyz
pbilat
pcomb
pcompos
pcond
pcwarp
pdelta
pdfblur
pexpand
pextrem
pfilt
pflip
pgblur
phisto
pinterp
pkgBSDF
plot4
plotin
pmapdump
pmblur
pmblur2
pmdblur
protate
psign
psketch
psmeta
psort
pvalue
ra_bmp
ra_gif
ra_hexbit
ra_pfm
ra_pict
ra_ppm
ra_ps
ra_rgbe
ra_t16
ra_t8
ra_tiff
ra_xyze
rad
rad2mgf
raddepend
ran2tiff
ranimate
ranimove
rcalc
rcollate
rcontrib
replmarks
rfluxmtx
rhcopy
rhinfo
rholo
rhoptimize
rhpict
rlam
rlux
rmake
rmtxop
rpict
rpiece
rsensor
rtcontrib
rtpict
rtrace
rttree_reduce
rview
rvu
tabfunc
testBSDF
tmesh2rad
total
trad
ttyimage
vinfo
vwrays
vwright
wrapBSDF
x11meta
xform
xglaresrc
ximage
xshowtrace
xyzimage

WHO WOULD SIGN UP FOR SUCH TORTURE?

Early Adopters

- UC Berkeley Architecture Dept.
- LESO Researchers @ EPFL
- Rob Shakespeare @ Indiana Univ.

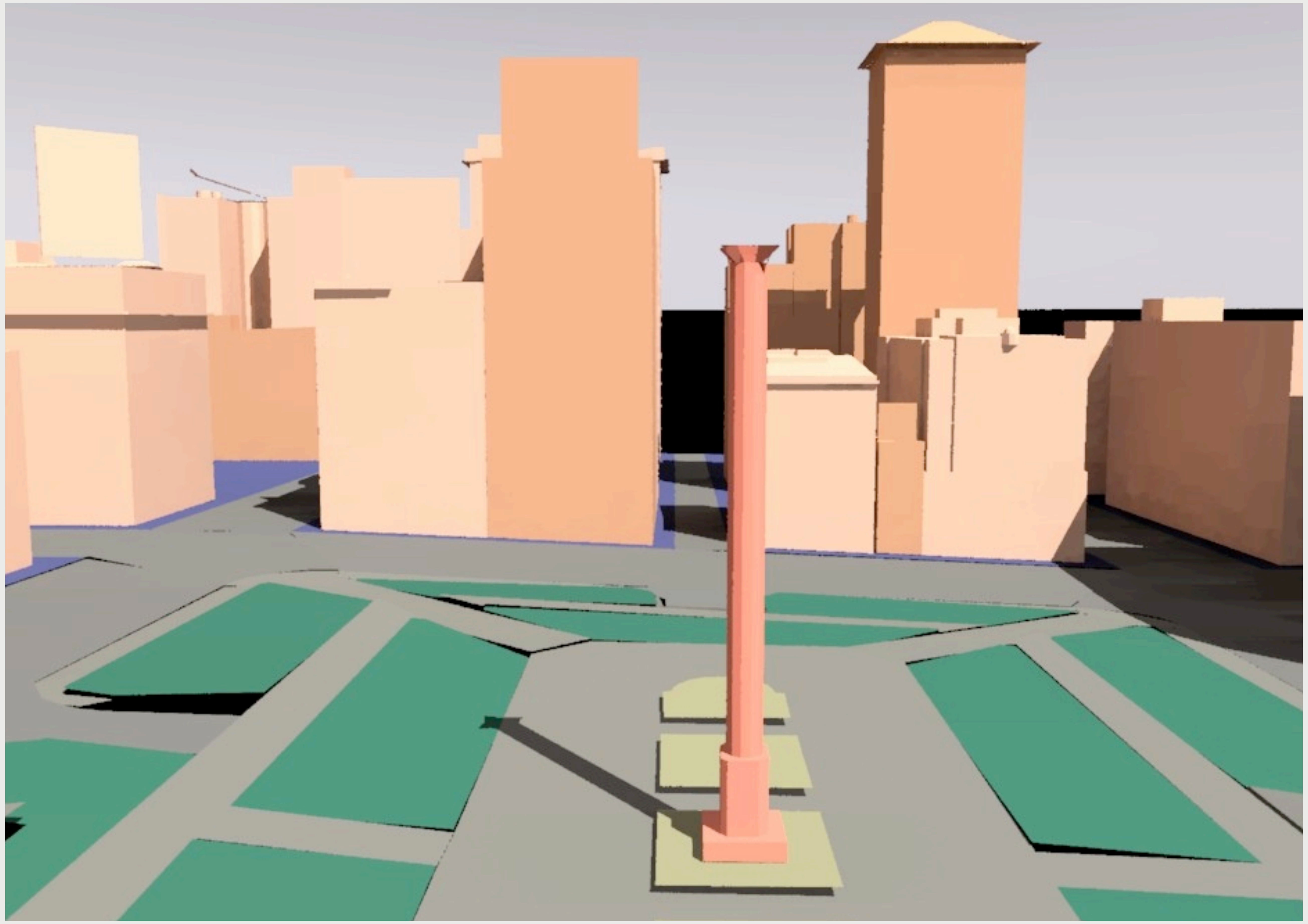


Chas Ehrlich

WHO WOULD SIGN UP FOR SUCH TORTURE?

Early Adopters

- UC Berkeley Architecture Dept.
- LESO Researchers @ EPFL
- Rob Shakespeare @ Indiana Univ.



Kevin Matthews

WHO WOULD SIGN UP FOR SUCH TORTURE?

Early Adopters

- UC Berkeley Architecture Dept.
- LESO Researchers @ EPFL
- Rob Shakespeare @ Indiana Univ.

Chas Ehrlich

WHO WOULD SIGN UP FOR SUCH TORTURE?

Early Adopters

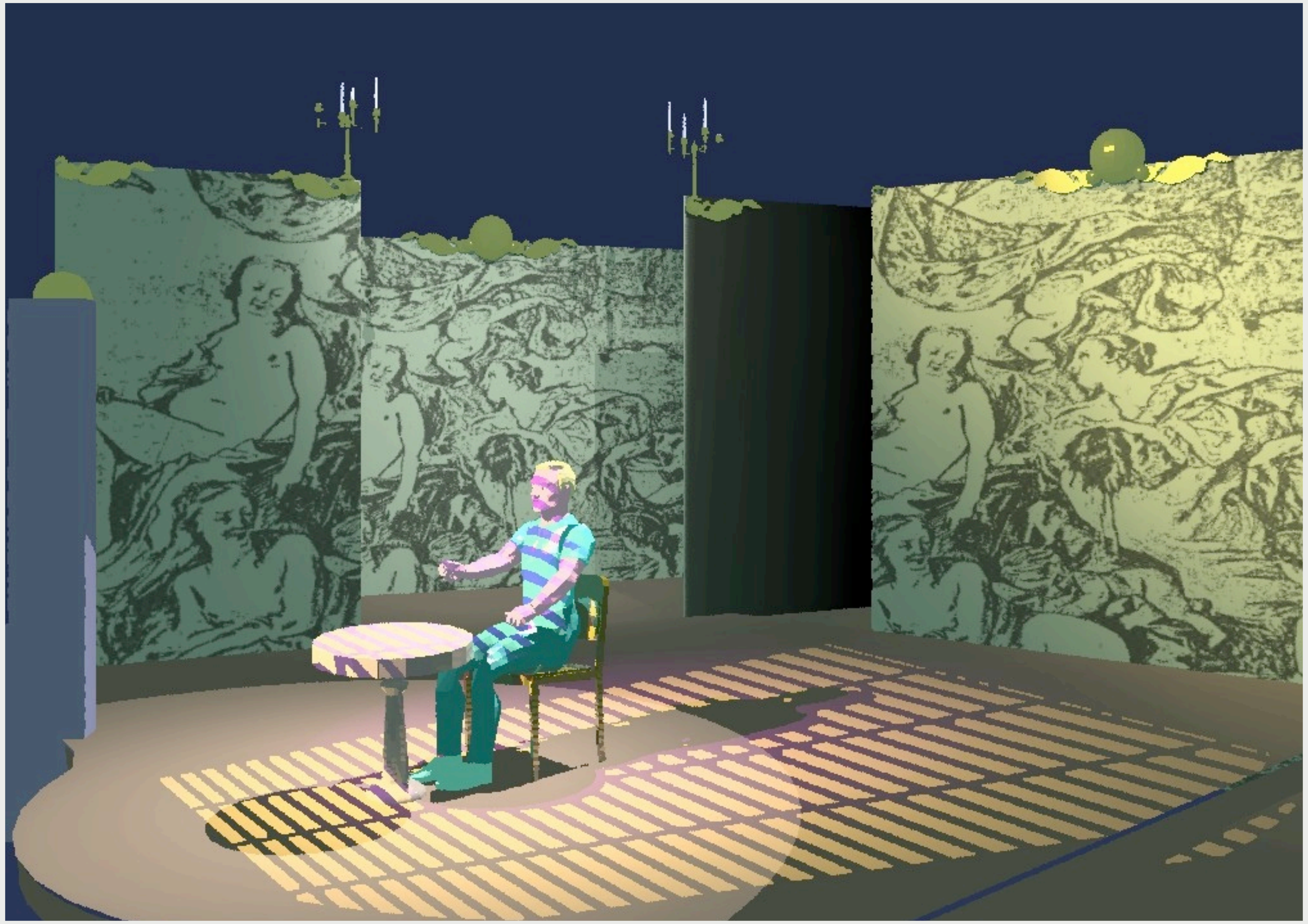
- UC Berkeley Architecture Dept.
- LESO Researchers @ EPFL
- Rob Shakespeare @ Indiana Univ.

Raphaël Compagnon

WHO WOULD SIGN UP FOR SUCH TORTURE?

Early Adopters

- UC Berkeley Architecture Dept.
- LESO Researchers @ EPFL
- Rob Shakespeare @ Indiana Univ.



Rob Shakespeare

WHO WOULD SIGN UP FOR SUCH TORTURE?

Early Adopters

- UC Berkeley Architecture Dept.
- LESO Researchers @ EPFL
- Rob Shakespeare @ Indiana Univ.

Rob Shakespeare

QUANTITATIVE VALIDATION

- LBNL's initial numerical validation compared electric and simple daylight calculations against existing tools

- More advanced validation against real-world measurements in full-scale model conducted by John Mardaljevic using sky scanner

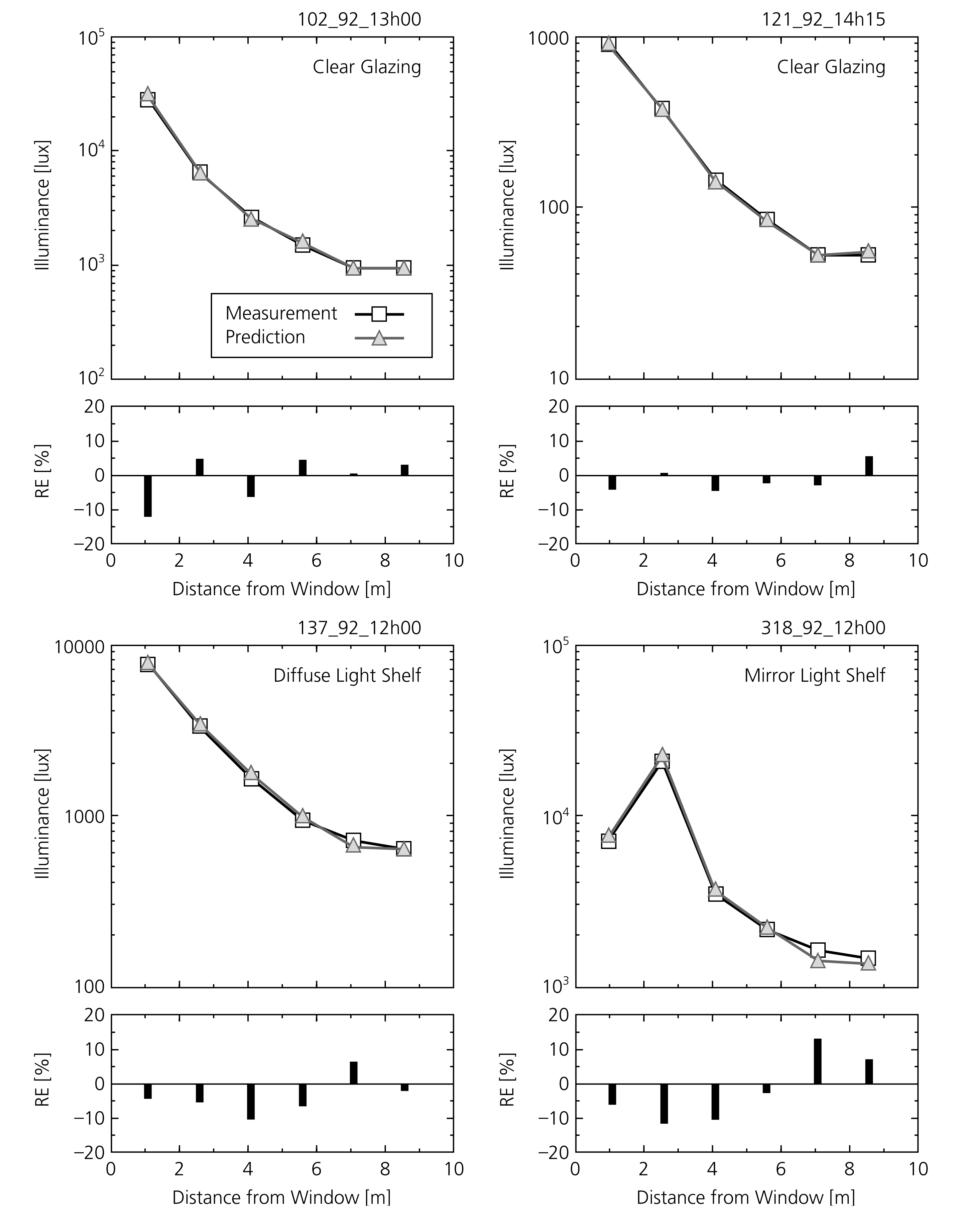


Figure 1.1 An experimental comparison between *Radiance* calculations and real measurements under daylight conditions [Mar95].

QUALITATIVE VALIDATION

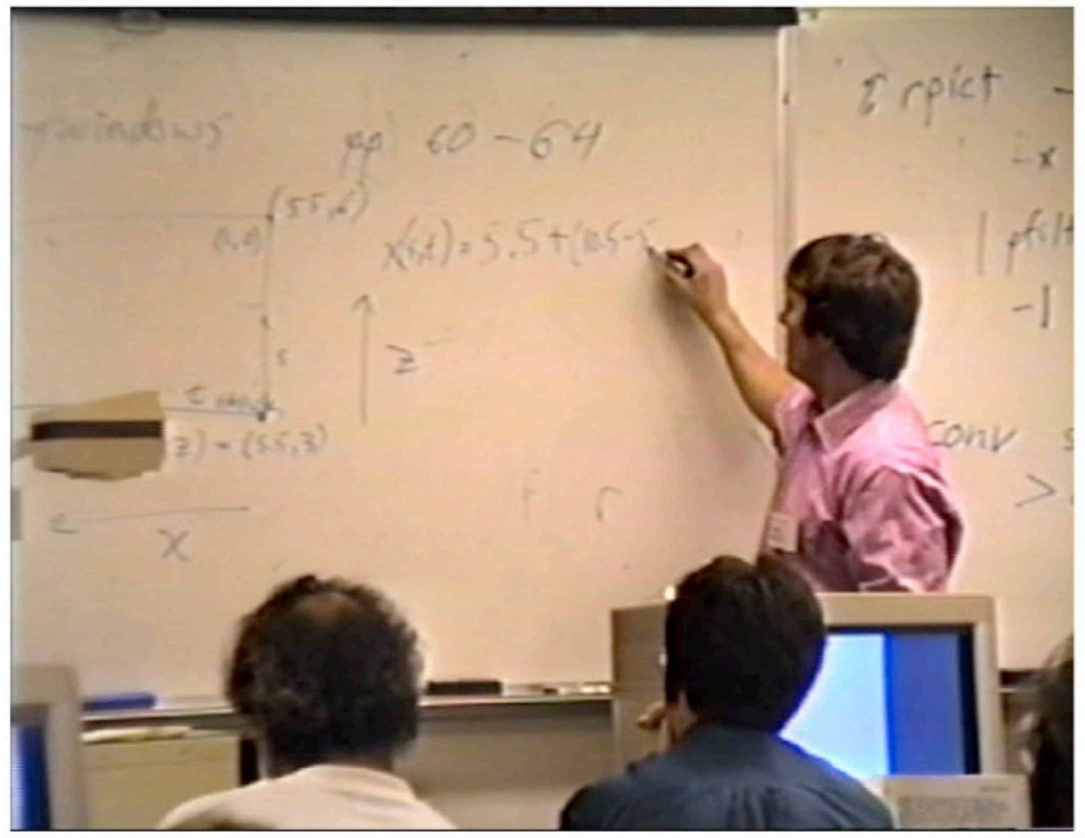
Beyond demonstrating the expected result on simple models, we aspired to simulate the built environment in all its complexity

QUALITATIVE VALIDATION

Anat Grynberg helped measure and model properties for the conference room at LBL in this comparison

First *Radiance* Workshop
(February 1991)

Next workshop did not
happen until 2002…



CHALLENGES WE FACED

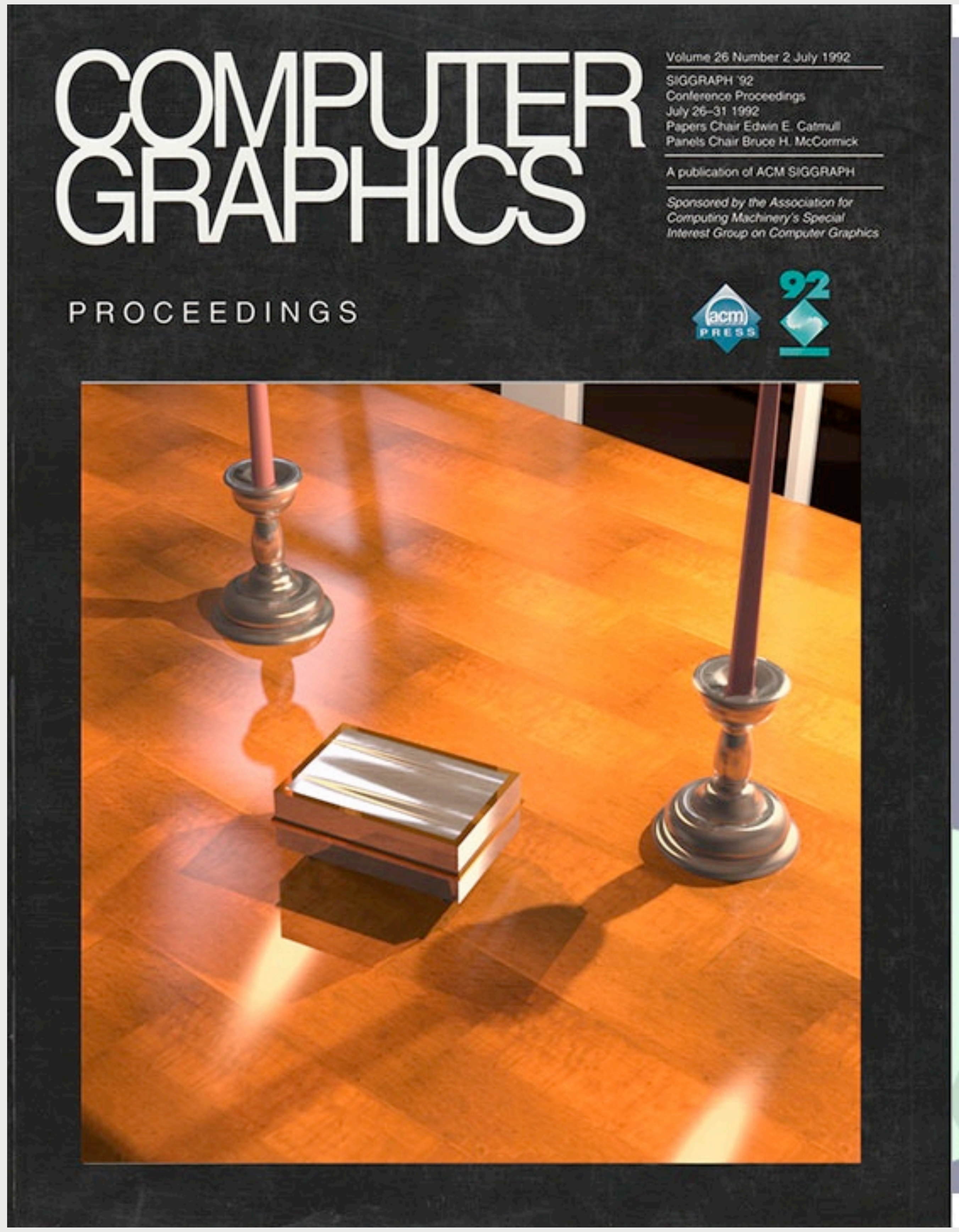
1. Intensity of daylight and importance of windows and interreflections
2. Characterizing and modeling material interactions
3. Evaluation of discomfort and disability glare
4. Calculation time and interaction
5. Documentation, outreach and education

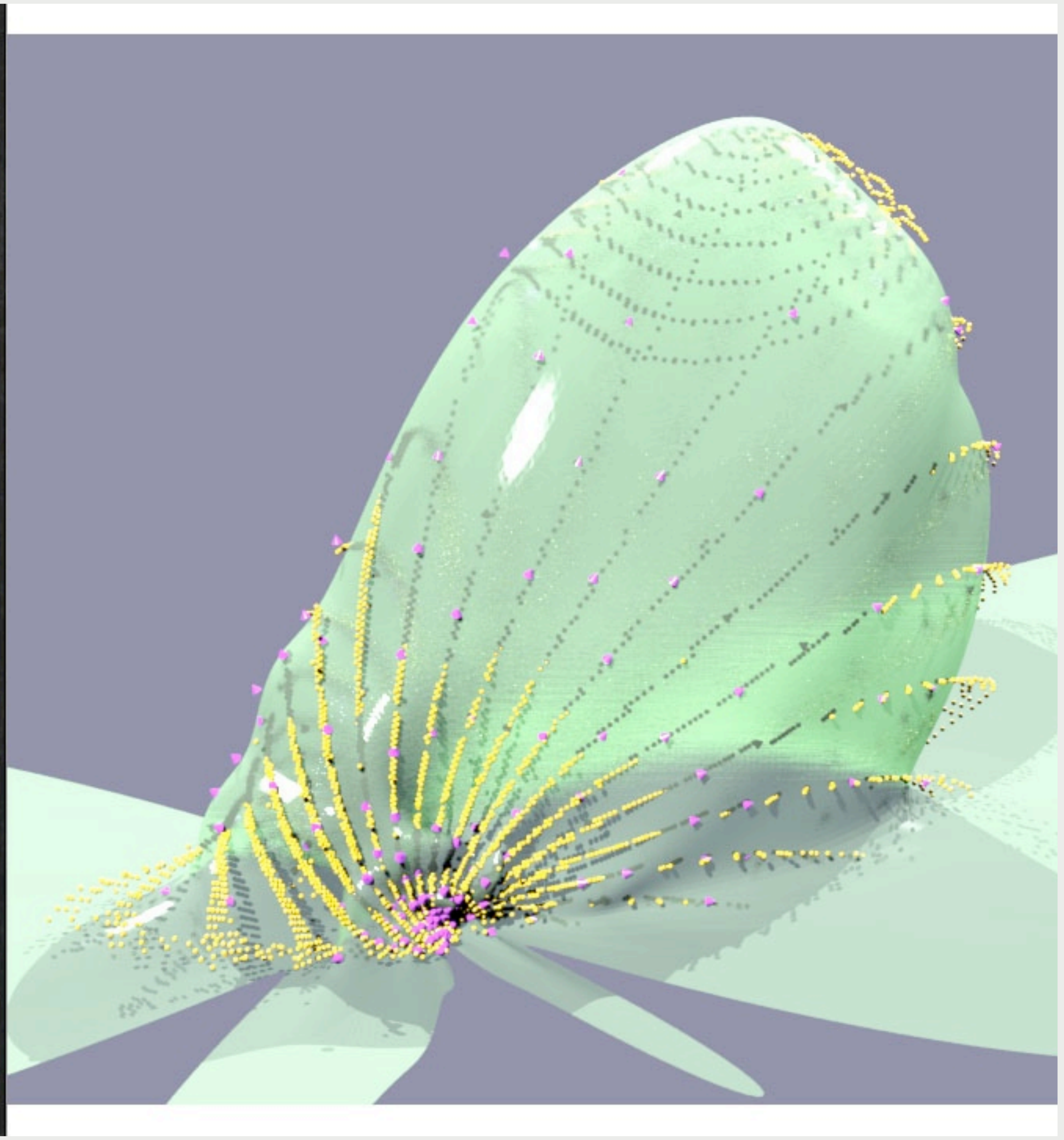
CHALLENGE RESPONSE (1)

- Intensity of daylight and importance of windows and interreflections
  - Secondary and virtual sources

CHALLENGE RESPONSE (2)

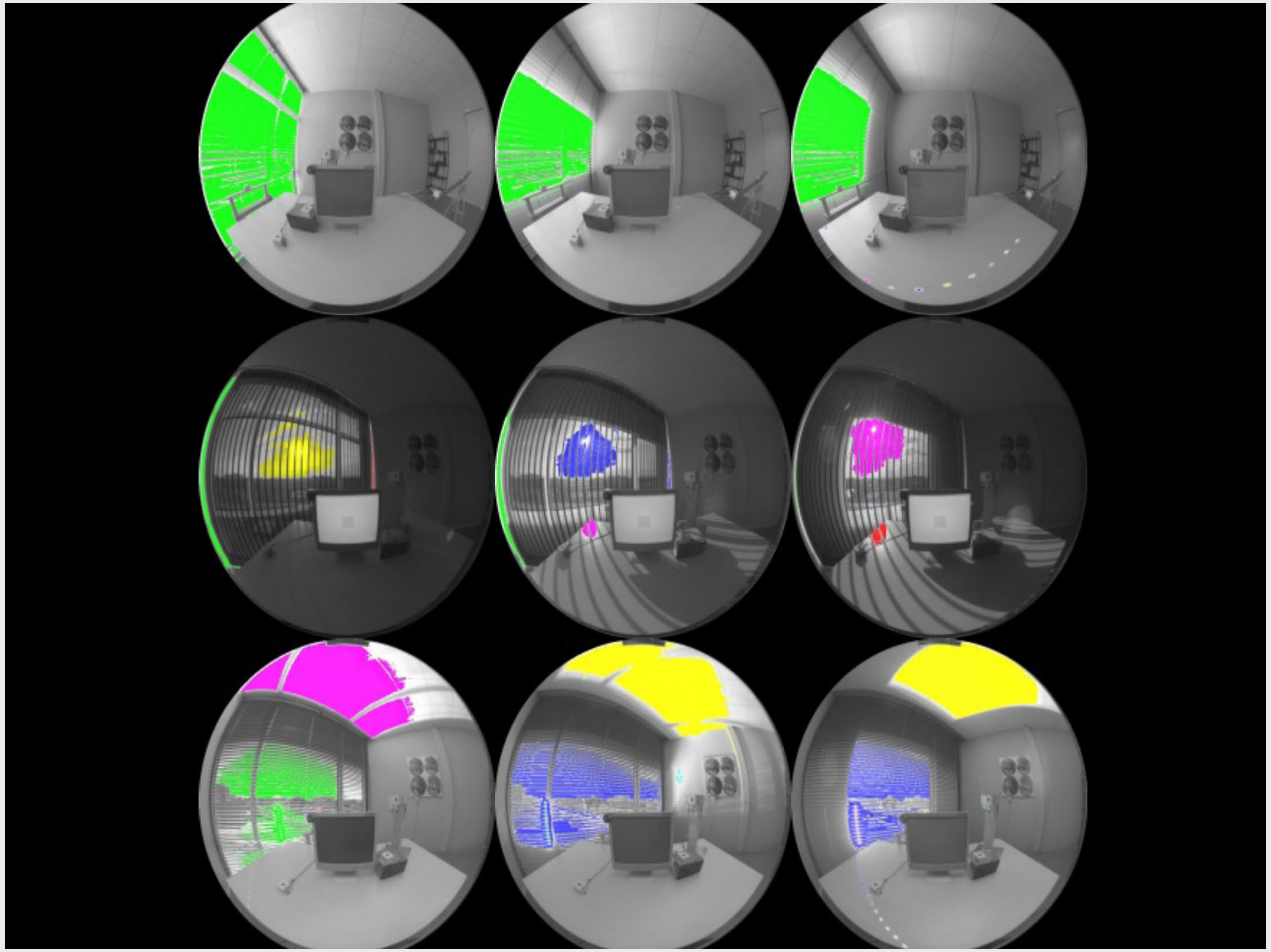
- Characterizing and modeling material interactions
    - BSDF measurements and data-driven materials





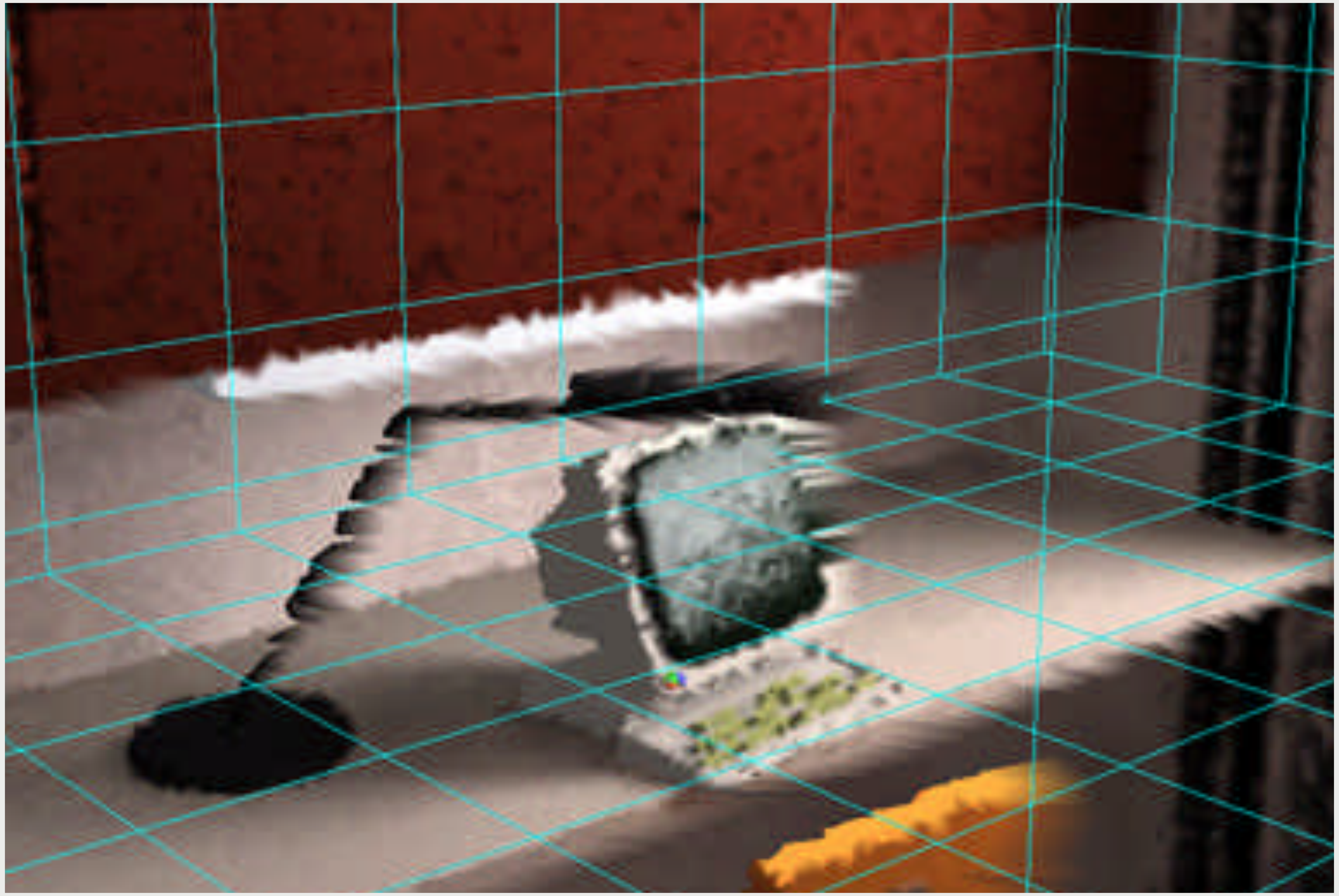
CHALLENGE RESPONSE (3)

- Evaluation of discomfort and disability glare
  - Findglare, evalglare (Jan Wienold)



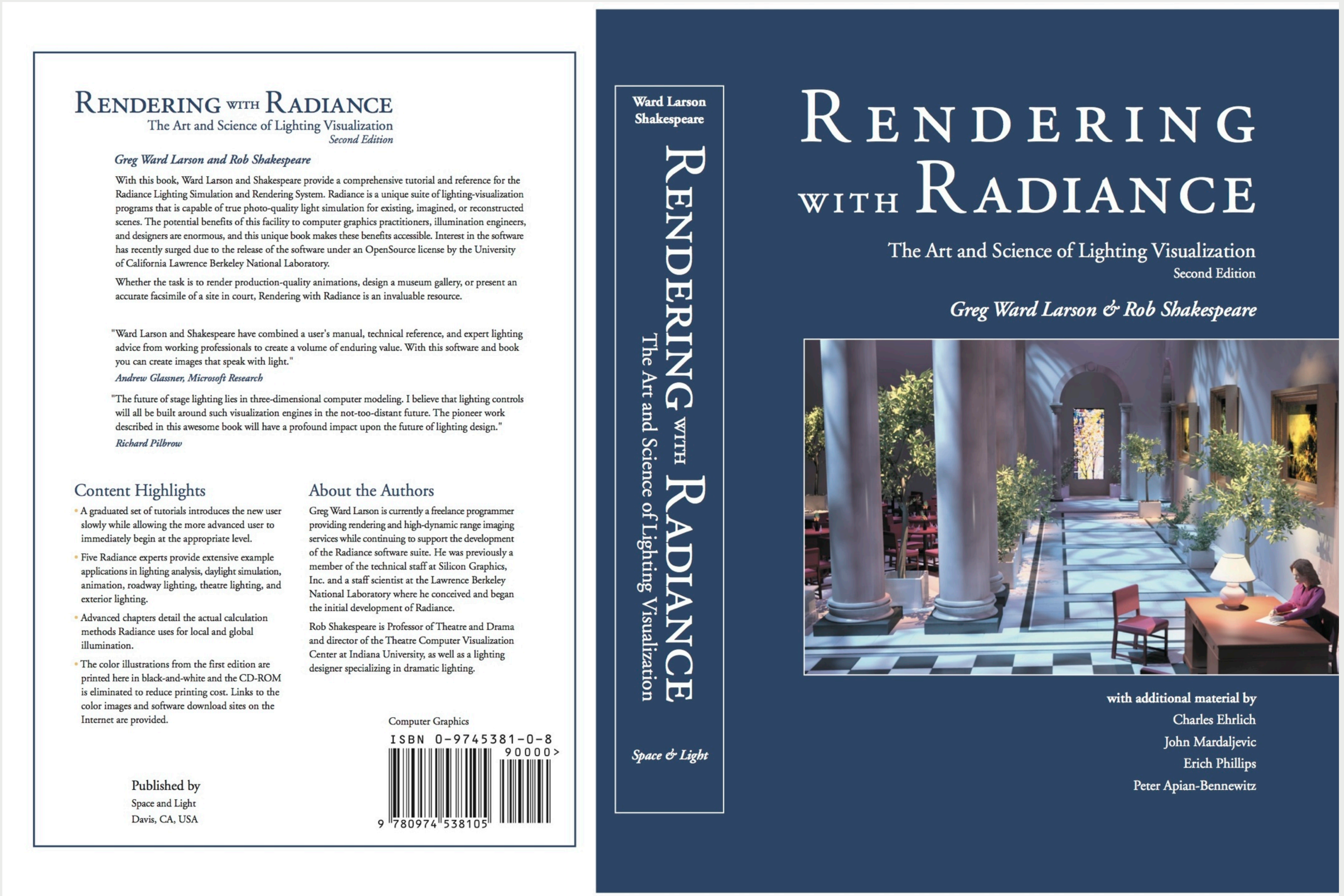
CHALLENGE RESPONSE (4)

- Calculation time and interaction
  - Holodeck, GUIs, annual simulation methods



CHALLENGE RESPONSE (5)

- Documentation, outreach and education
  - Mailing lists, textbook, tutorials, website, and workshops

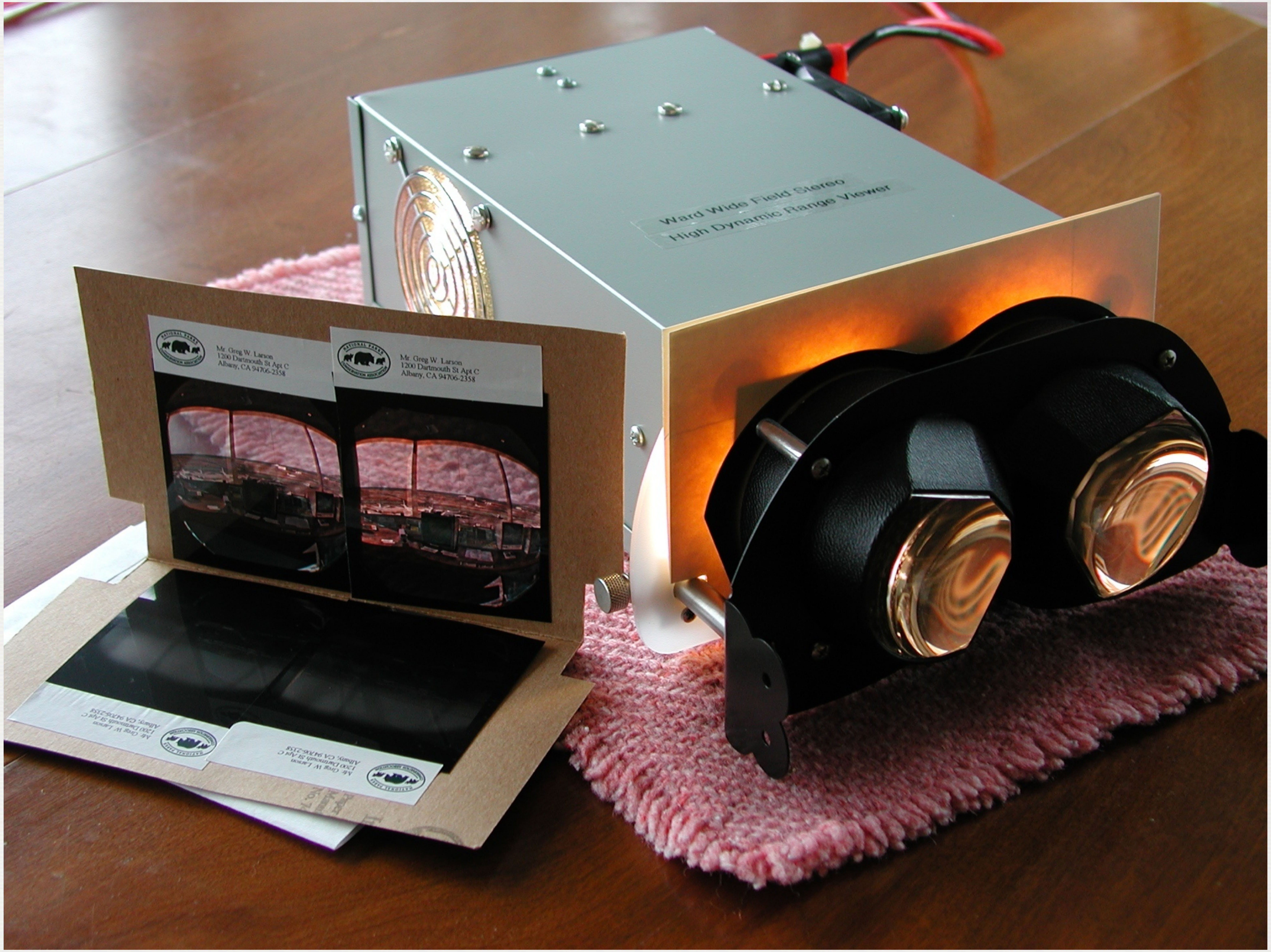


HDR IMAGING AS OUTGROWTH

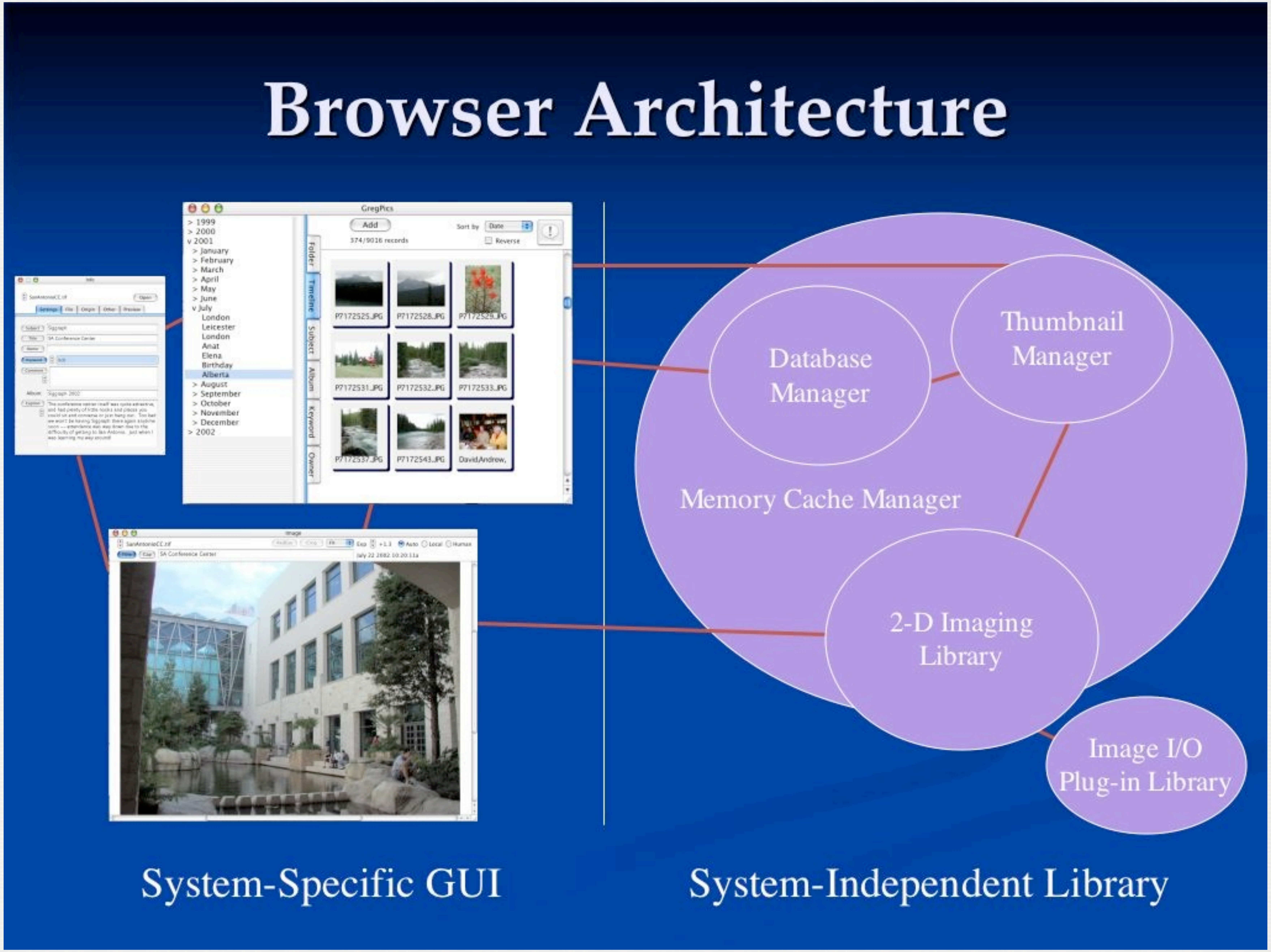
- Tone-mapping methods (**pcond**)
- LogLuv TIFF & JPEG-HDR image formats
- Image-based lighting (Paul Debevec)
- *Photosphere* application & photometry
- HDR viewer & Brightside/Dolby display

Image-based Lighting:
Fiat Lux
by Paul Debevec

HDR Stereo Viewer
(inspired dual-
modulation display)



Photosphere
HDR Image Builder / Browser



RADIANCE TODAY

- Community of 1000 active users (according to Discourse server)
- At least a dozen softwares that use or build on *Radiance* simulation engine
- Scores of Ph.D. theses employed *Radiance* calculations, more each year
- 17 *Radiance* annual workshops (Raphaël Compagnon started in 2002)
  - Just met in Loughborough; next year will be in NYC
- Active research on BSDF representation & sharing, annual simulations

CONTRIBUTORS

- Jean-Jacques Delaunay - Perez sky model (gendaylit)
- Peter Apian-Bennewitz - *rshow* & radiance-online.org
- Georg Mischler - *rayfront*
- Christoph Reinhart - DAYSIM
- John Mardaljevic - annual daylight simulation and validation
- Roland Schregle - photon-mapping additions to *Radiance*

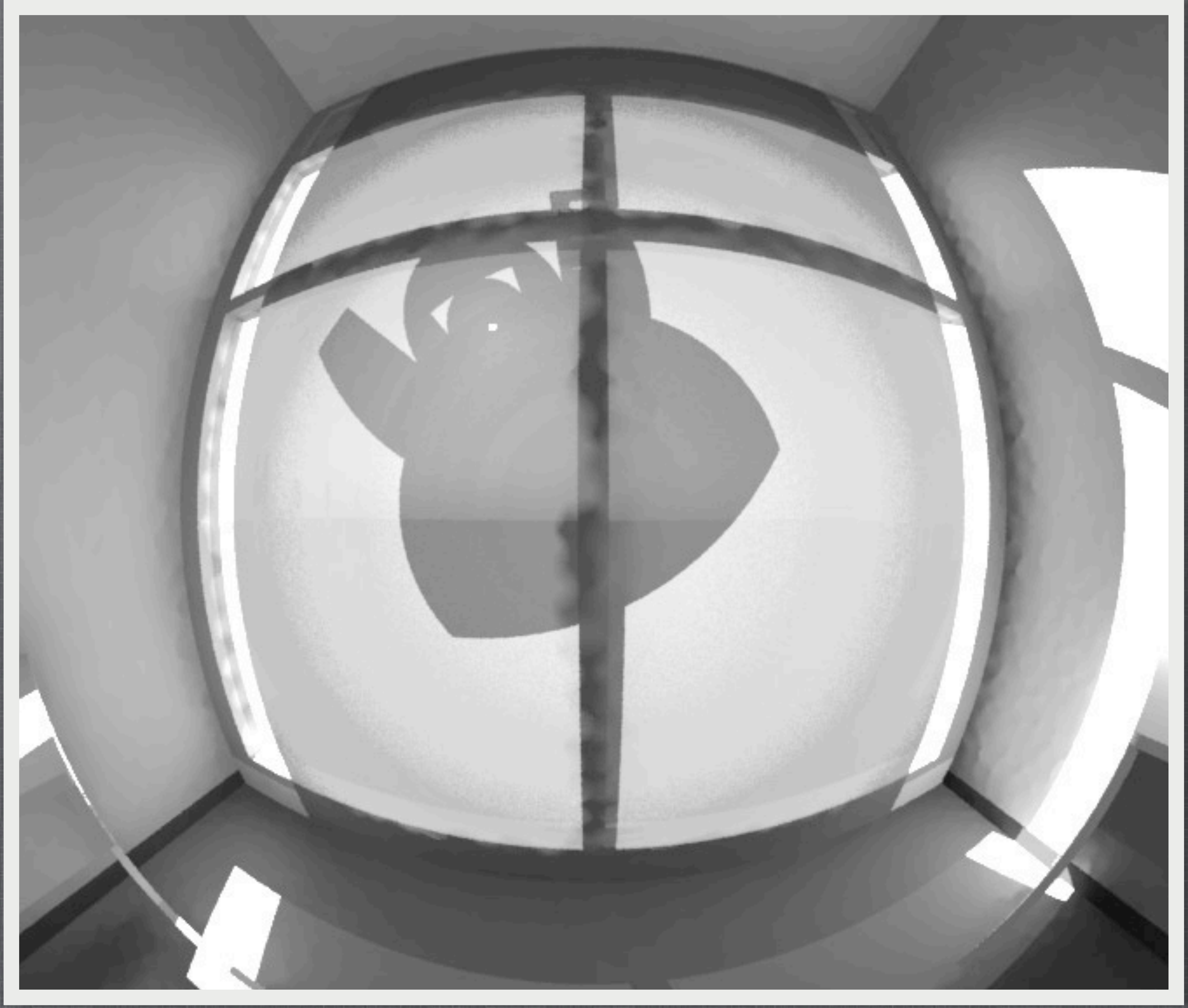
CONTRIBUTORS (CONT'D)

- Carsten Bauer - *Radzilla*
- Andrew Marsh - ECOTECT
- Zack Rogers - SPOT
- NREL & Rob Guglielmetti - *OpenStudio* measure
- Jan Wienold - **evalglare**
- Mostapha Sadeghipour Roudsari - *Ladybug & Honeybee*
- Nathaniel Jones - *Accelerad*

...plus numerous others

RADIANCE'S FUTURE

- Partly up to you — DOE funding is a perennial problem
- Code is mature and stable, but no guarantee of continued longevity
- Work is beginning this year on automated regression testing
  - Reduces primary author's importance as gate-keeper
  - Theoretically, testing will ensure validity of new modifications
- Guaranteed job security: there are always bugs that will surprise!



Francesco Anselmo
Dan Glaser
Alan Chalmers
Sarith Subramaniam
Alstan Jakubiec
Lars Grobe
Richard Mistrick
Jean-Louis Scartezzini
David Geisler-Moroder
Andy McNeil
Mehlika Inanici
Axel Jacobs
Holly Rushmeier
Chris Humann
Germán Molina

THANK YOU.
AND MORE...

Rob Guglielmetti
Giulio Antonutto
Bob Clear
Judy Lai
Steve Selkowitz
Eleanor Lee
Paul Heckbert
Dan Fuller
Kera Lagios
Saba Rofchai
Luisa Brotas
Cindy Larson
Alstan Jakubiec
Taoning Wang
Kevin Van Den Wymelenberg
Jennifer Schuman


John Mardaljevic


Martin Moeck

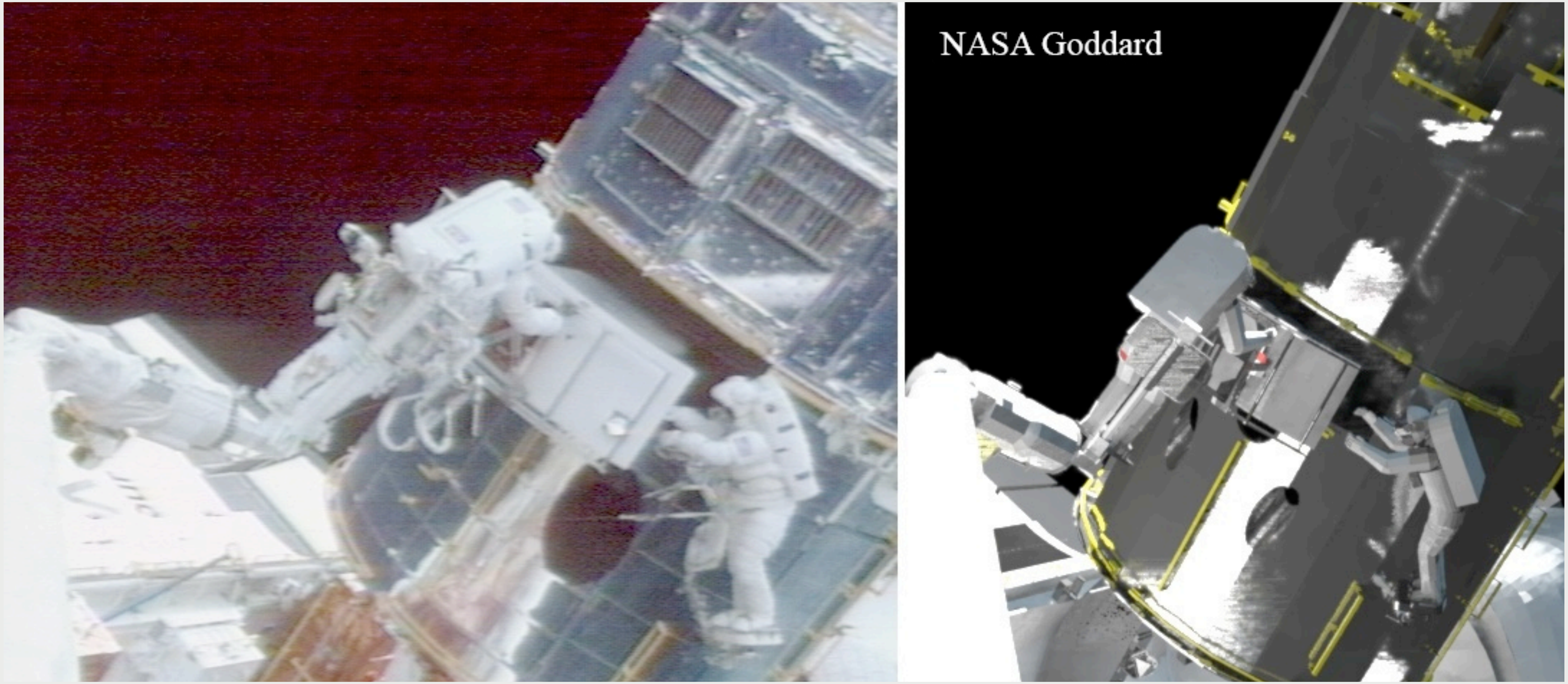

Matiu Carr


Reuben McFarland & Scott Routen

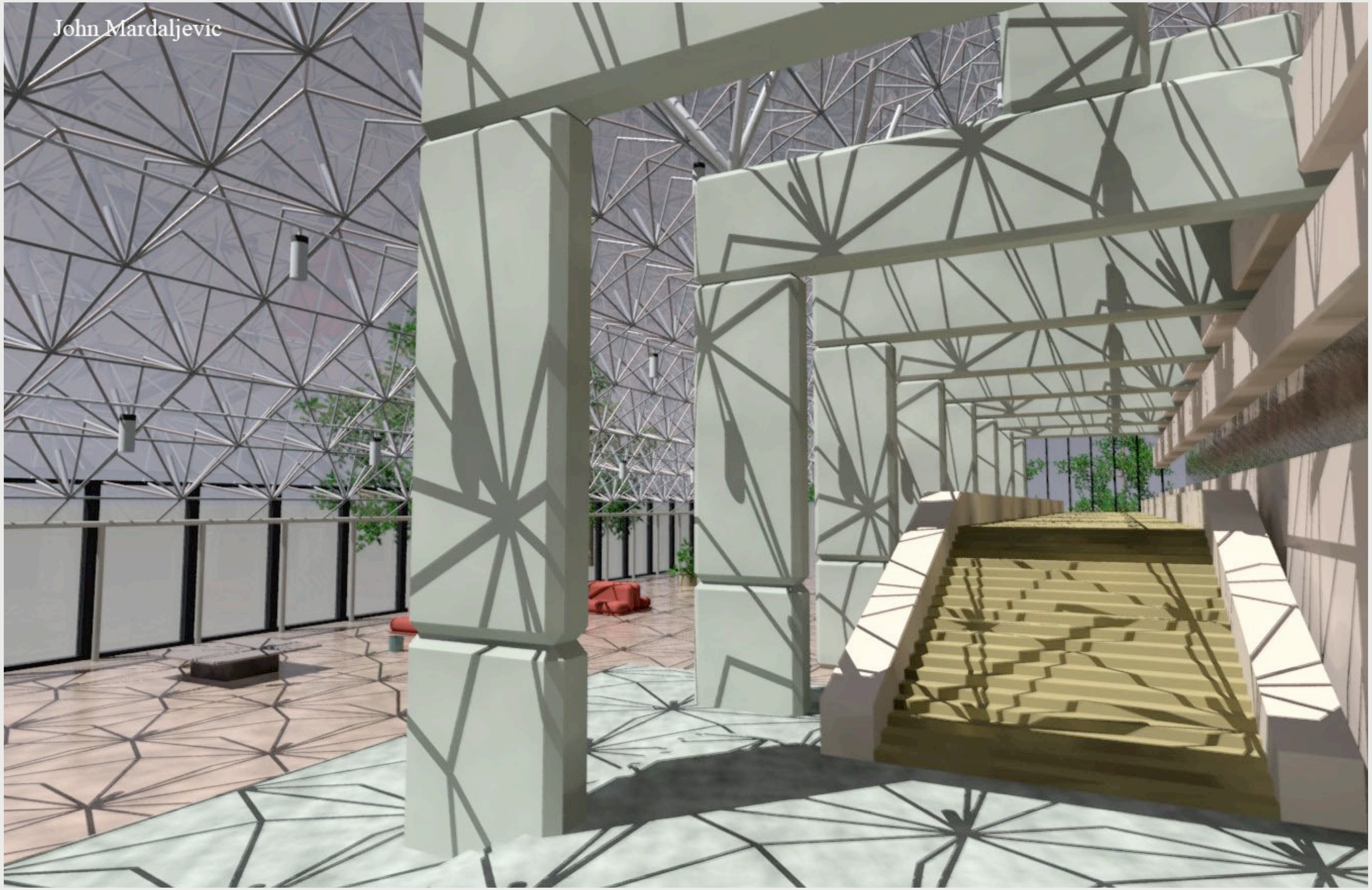

Peter Apian-Bennewitz
FhG ISE


Chas Ehrlich


Jack deValpine


NASA Goddard


Martin Moeck


John Mardaljevic


Deanan Dasilva


Clement Pelou


Andrew Sedgwick


Veronica Sundstedt & Patrick Ledda