21st International
Radiance Workshop
in Innsbruck/Austria

28.-31. Aug. 2023

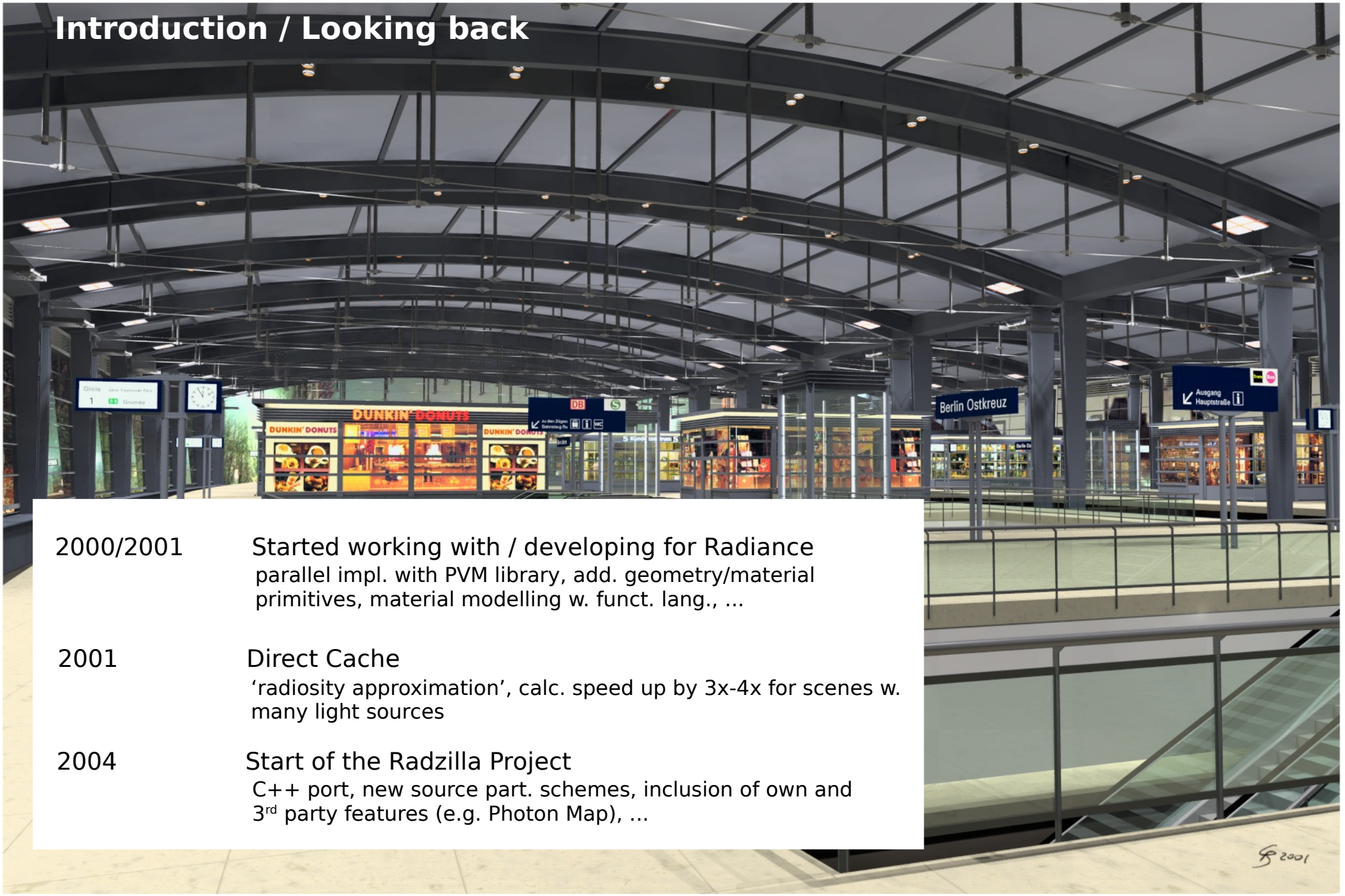
Radzilla 3.33

Carsten Bauer

Introduction / Looking back

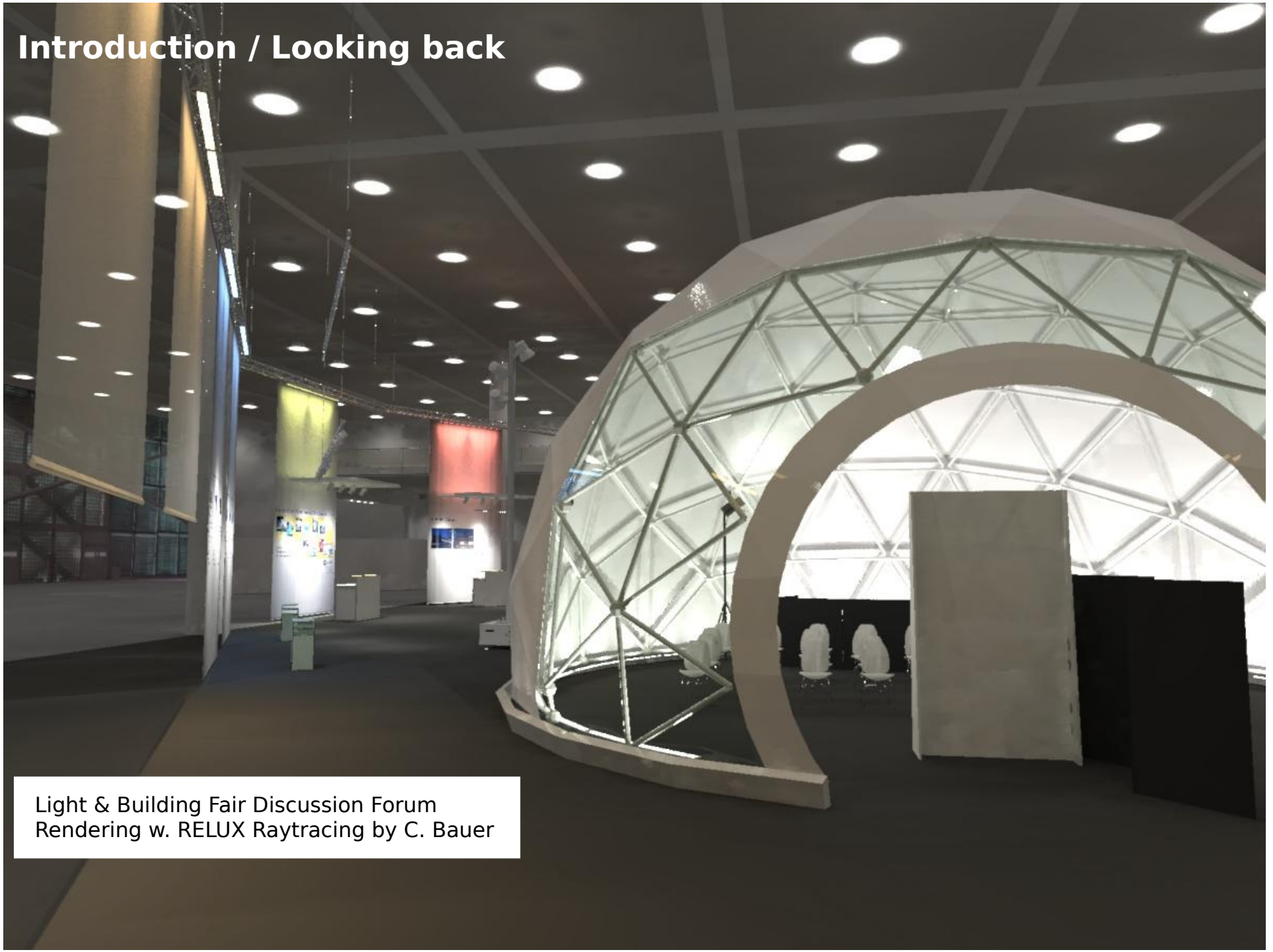
Berlin Ostkreuz Station
Lighting Design: E. Schlaefle
Rendering C. Bauer (2001)

Introduction / Looking back



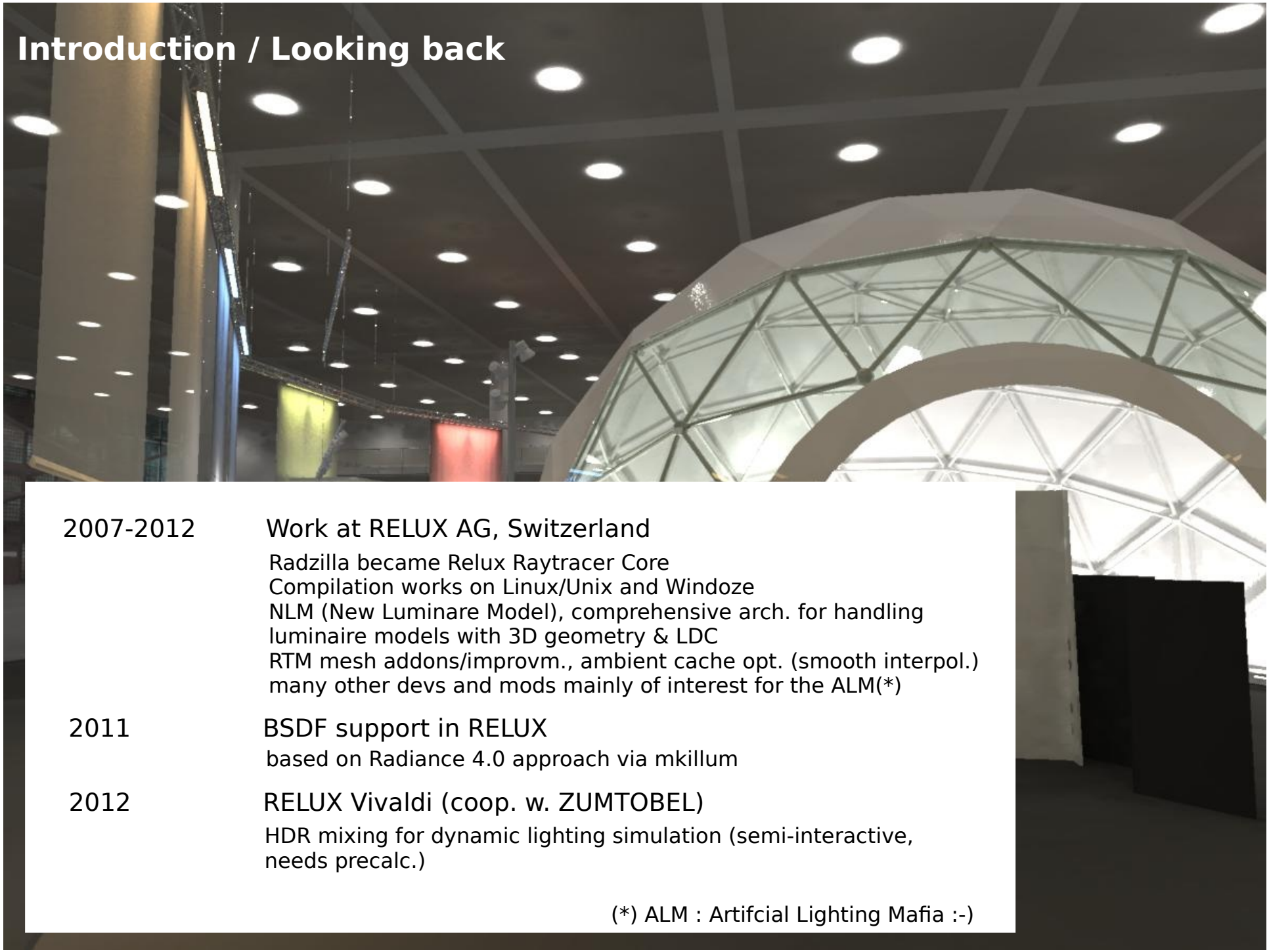
2000/2001	Started working with / developing for Radiance parallel impl. with PVM library, add. geometry/material primitives, material modelling w. funct. lang., ...
2001	Direct Cache ‘radiosity approximation’, calc. speed up by 3x-4x for scenes w. many light sources
2004	Start of the Radzilla Project C++ port, new source part. schemes, inclusion of own and 3rd party features (e.g. Photon Map), ...

Introduction / Looking back



Light & Building Fair Discussion Forum
Rendering w. RELUX Raytracing by C. Bauer

Introduction / Looking back



2007-2012	Work at RELUX AG, Switzerland
	Radzilla became Relux Raytracer Core Compilation works on Linux/Unix and Windoze NLM (New Luminare Model), comprehensive arch. for handling luminaire models with 3D geometry & LDC RTM mesh addons/improvm., ambient cache opt. (smooth interpol.) many other devs and mods mainly of interest for the ALM(*)
2011	BSDF support in RELUX
	based on Radiance 4.0 approach via mkillum
2012	RELUX Vivaldi (coop. w. ZUMTOBEL)
	HDR mixing for dynamic lighting simulation (semi-interactive, needs precalc.)

(*) ALM : Artifcial Lighting Mafia :-)

Introduction / Looking back

2013	Start of the Radzilla Interactive Raytracer project cf. following pages
2014/15	New parallel implementation for execution on Multicore CPUs based on Intel TBB library
2015/16	Work at Lucerne Univ. (HSLU) Radiance classic intermezzo, EvalDRC (Photon Map integration, CFS simulation framework based on rtcontrib)
~2019	First release of Radzilla Interactive Raytracer / RELUX Dynamic Planning preliminary state, indirect part still missing
2020-23	Development pause only maintenance/bugfixing continued
2023 ff.	?

Radzilla Interactive Raytracer

Key features:

- interactive interface for convenient working on lighting design scenes
- quick preview calculation for providing immediate feedback, although at lower precision/quality. This includes an image calculation in several steps (direct/indirect light, spec. refl.)
- option to perform fully accurate background calculation in a separate run
- runs on Linux and Windoze

Basic implementation aspects:

- comprehensive interface between Raytracer Core and RELUX GUI (command and data transfer via pipes, shared memory and files)
- enhanced/adapted geometry and material handling to accept and process edit actions
- monolithic program architecture (“all inclusive”, i.e. rzpict incorporates oconv, mkillum, pfilt, gensky, ...)
- parallel execution on Multicore CPUs with help of external library (Intel TBB)
- extensive data caching structures to handle the multi pass / several calc. steps approach
- inverted ambient cache algorithm with new interpolation scheme ***not yet ready :(***

Radzilla Image Formula

$$(B_d + B_a + ((B_{sd} + B_{sa})*TR + TRD_d + TRD_a)*sc\ )*TV + TVD_d + TVD_a + TVD_{sd} + TVD_{sa} \rightarrow \text{Image}$$

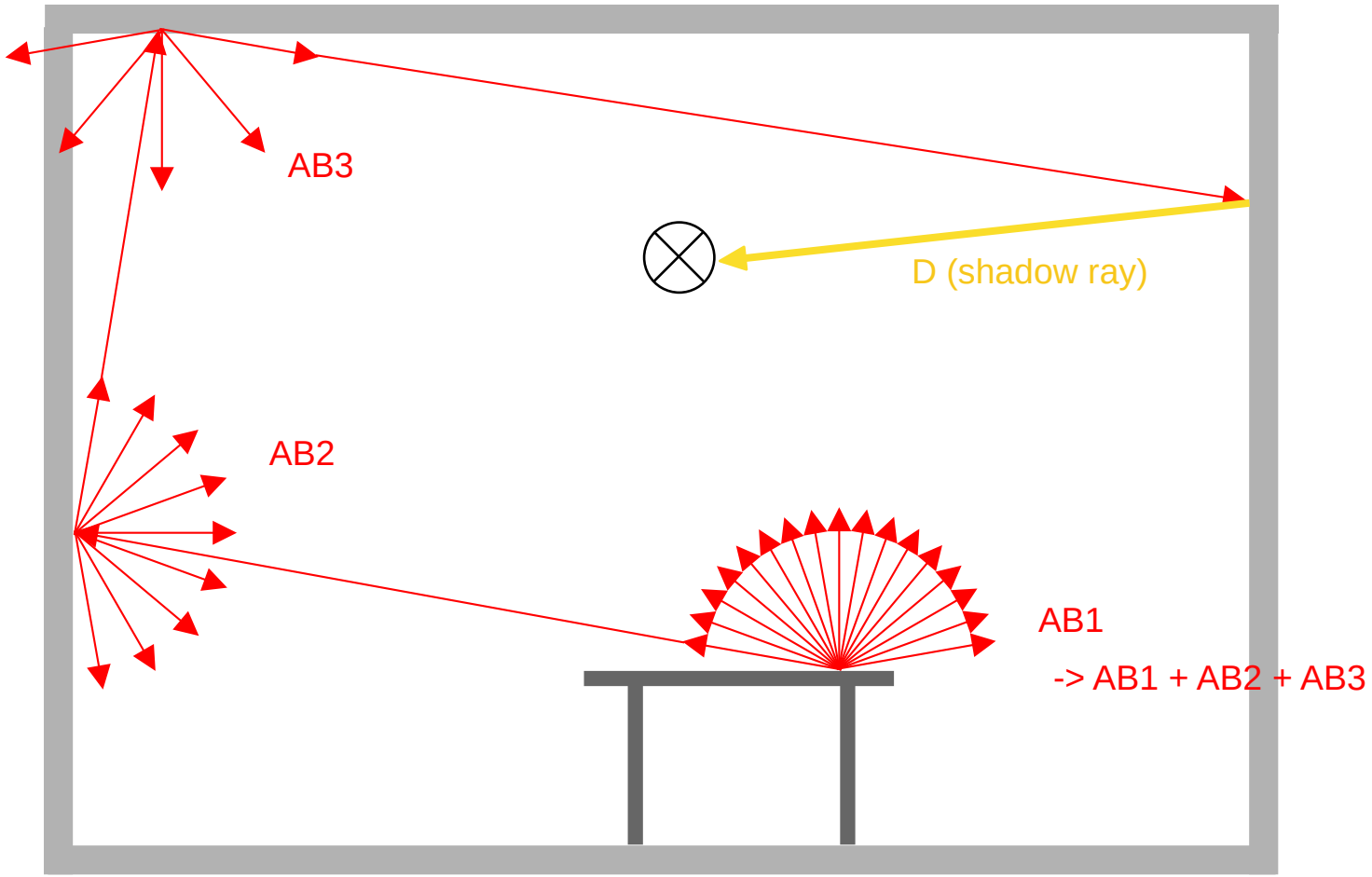
meaning of the letters:
- B : ‘base’, contribution from opaque objects
- T : influence of / contribution from transparent objects on the rays path (either as factor or - in comb. with D - as additional luminance)
- R : reflected direction / ray path
- V : view direction / ray path
- D : diffuse contribution of transparent objects (‘trans’ material)
-sc : specular reflection coefficient

Suffixes:
- d : direct light
- a : ambient/indirect light
- sd/sa : direct/ambient light on reflected objects

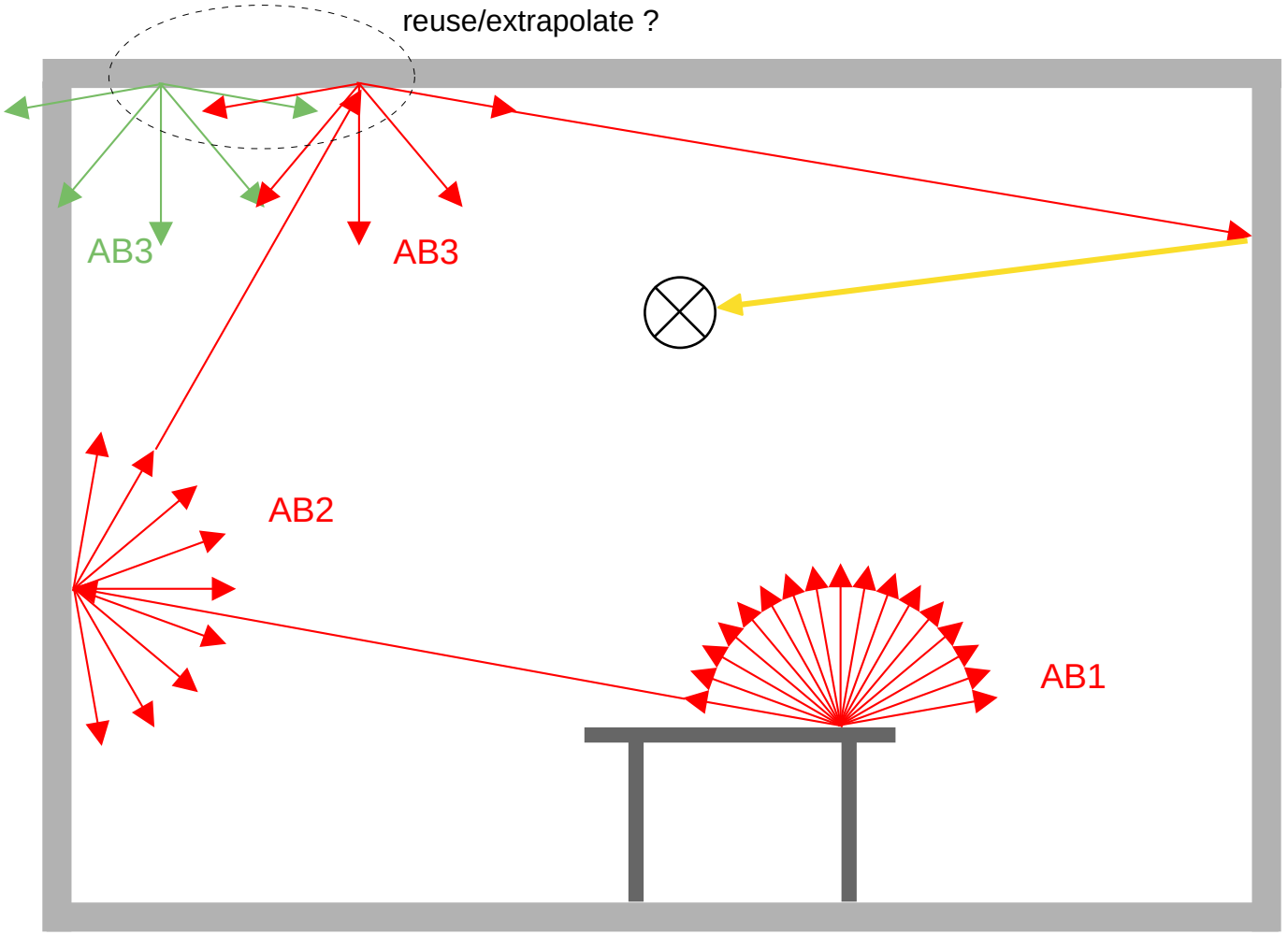
Approximation for only 1 specular reflection and separable ambient/specular contributions, resp. neglectable contribution of specular reflections to illumination

… sorting out the different contributions is left as exercise for the interested participant :)

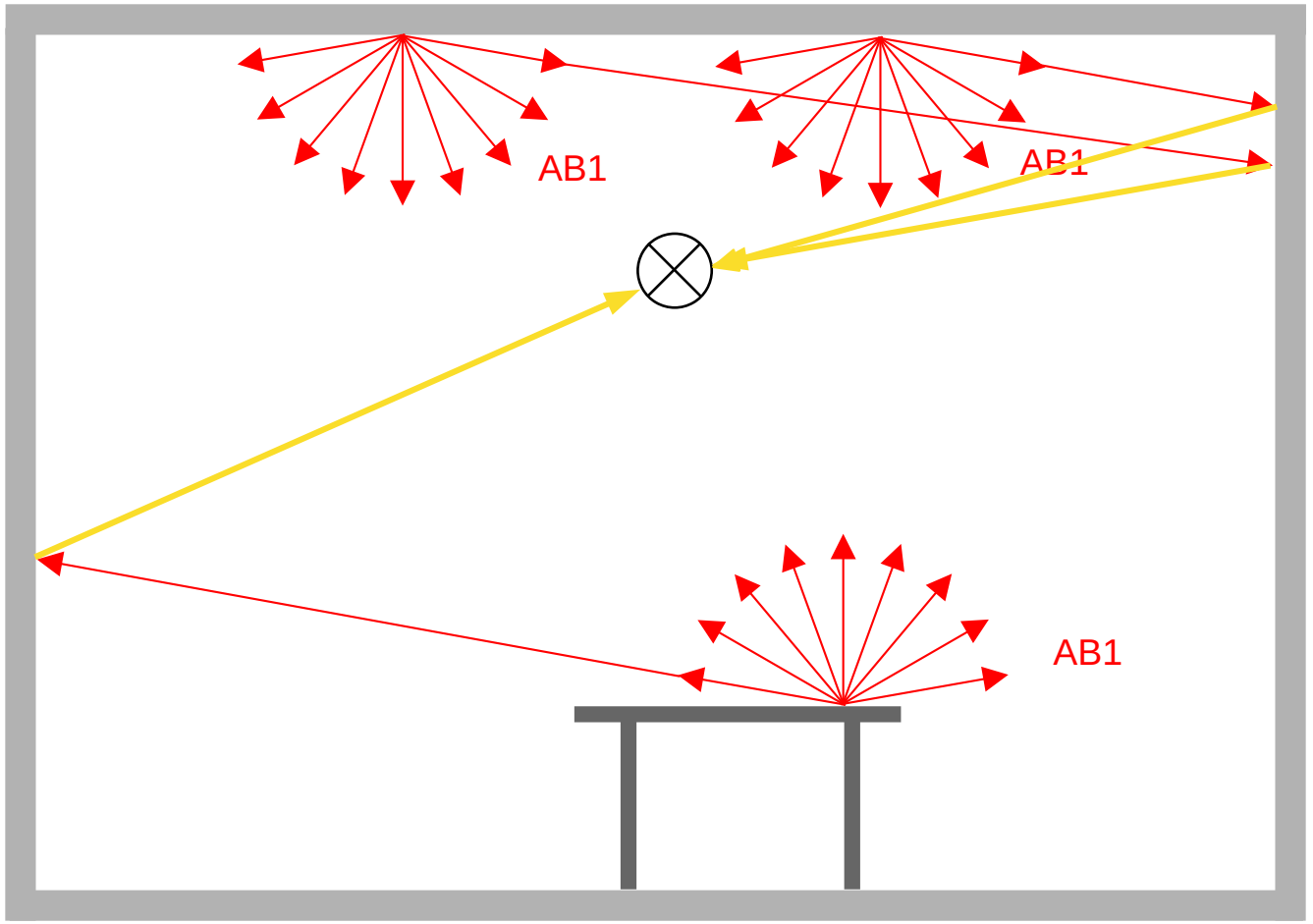
Radiance ambient algorithm



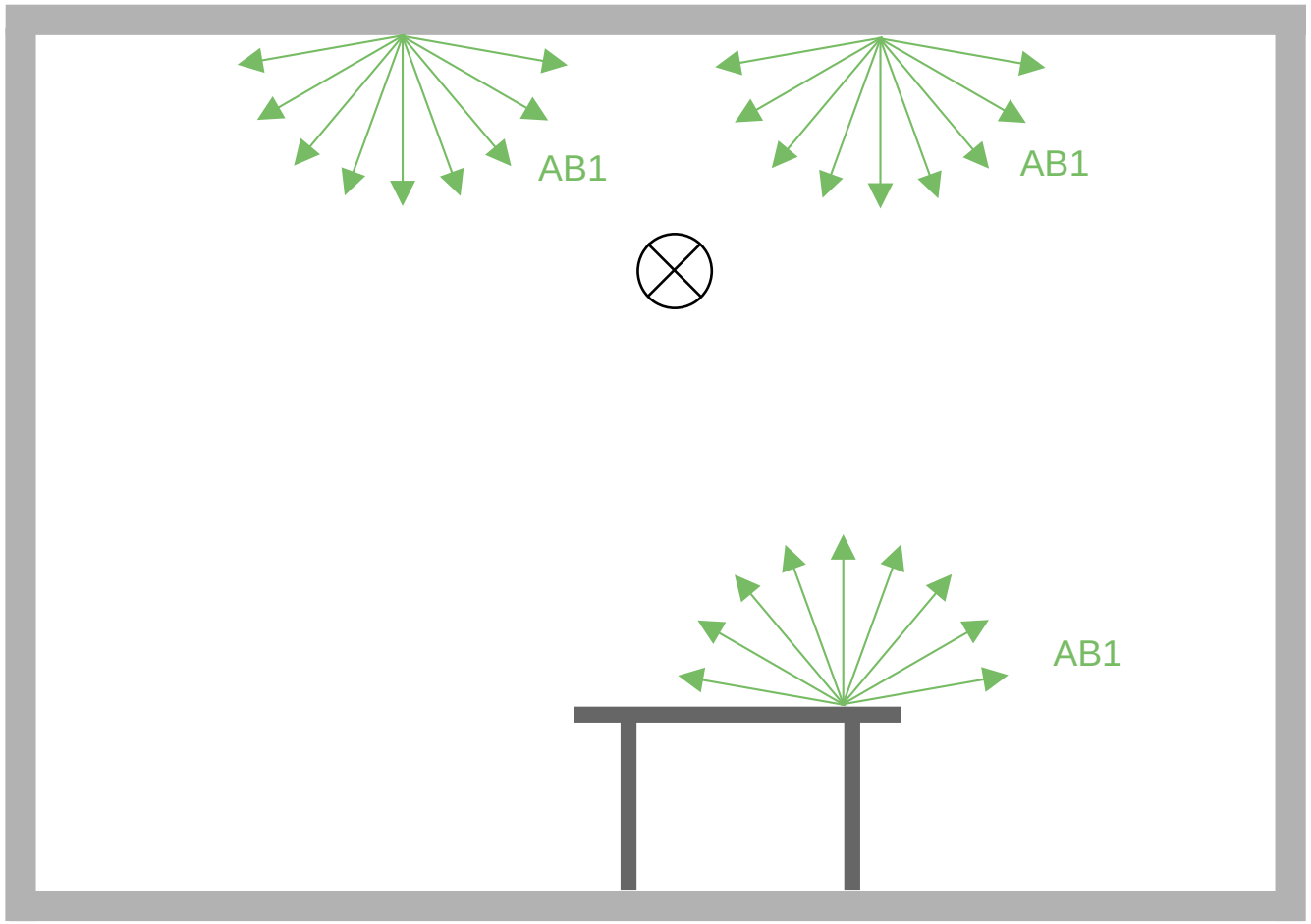
Radiance ambient algorithm



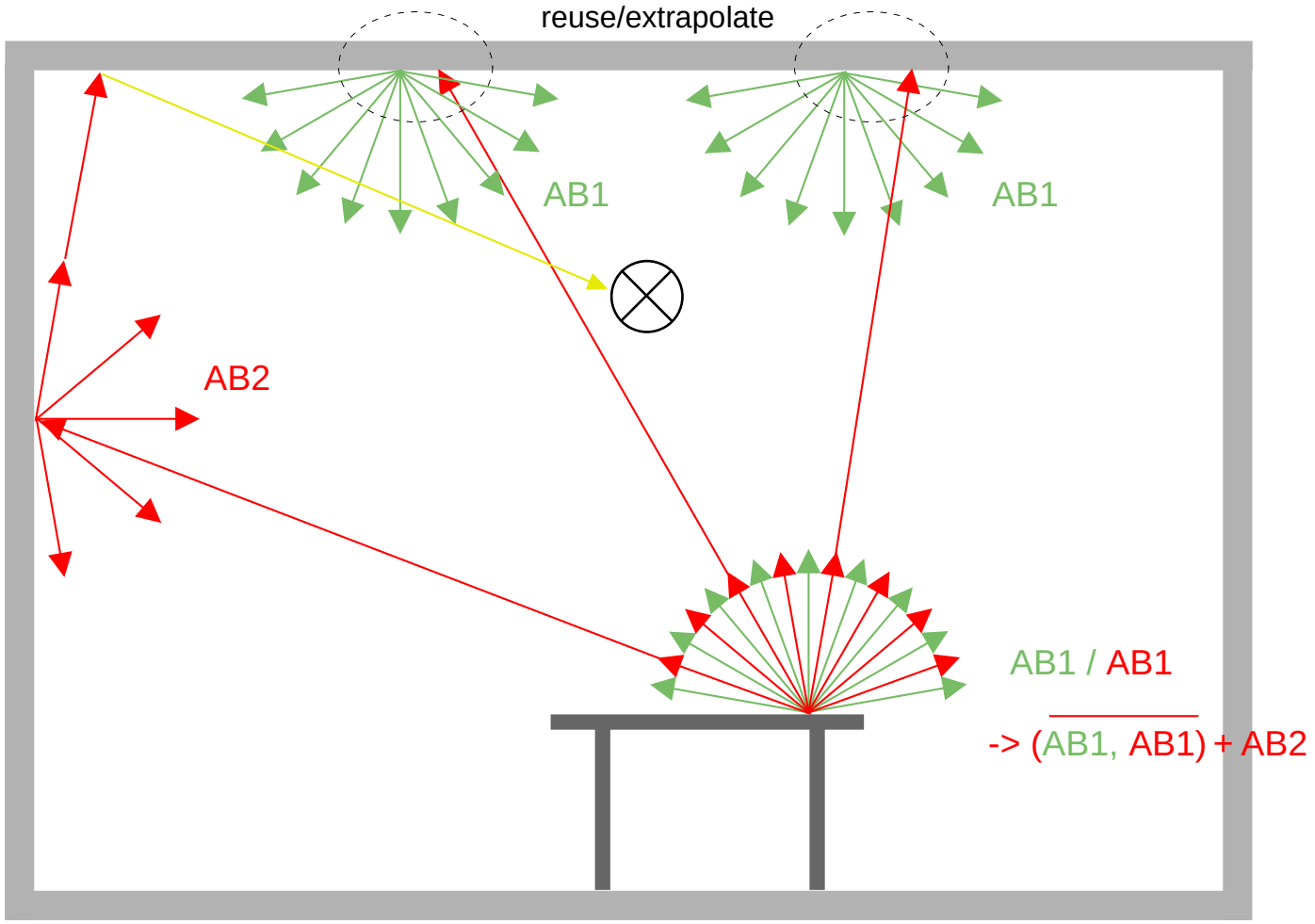
Radzilla inverted ambient algorithm (preview/interact. mode)



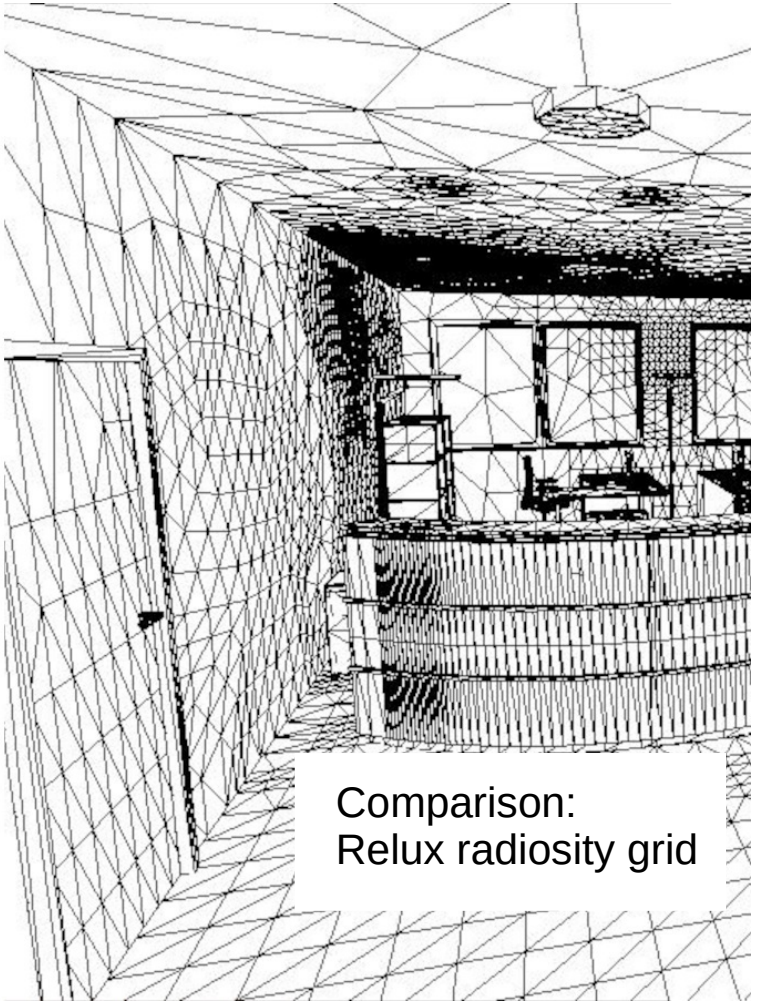
Radzilla inverted ambient algorithm (preview/interact. mode)



Radzilla inverted ambient algorithm (preview/interact. mode)



Radzilla ambient interpolation (preview/interact. mode)



Task yet to accomplish:
Organize / store ambient values in a mesh-like structure, comparable to a radiosity raster (ideally geometry-independent w. flexible resolution)

… left as exercise for the interested participant :)

This is the end.

I hope you enjoyed the show.

Any questions? No! Great, let's have coffee break!