The image as a virtual stage

Computational Photography

Derek Hoiem

Adapted from slides by Kevin Karsch, presented by Aditya Deshpande

Today

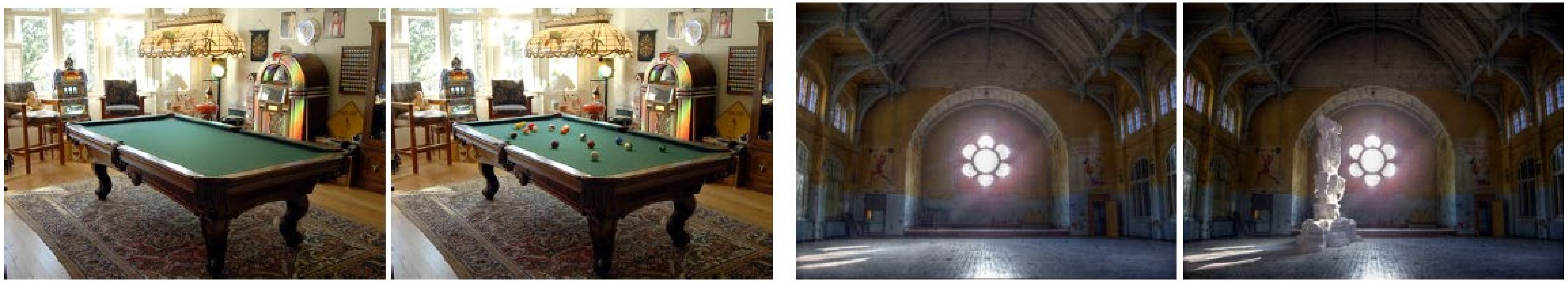
- Inserting objects into *legacy* photos
  - Uses single-view geometry and image-based lighting concepts

- Demo for using Blender

Rendering Synthetic Objects into Legacy Photographs

Kevin Karsch Varsha Hedau David Forsyth Derek Hoiem

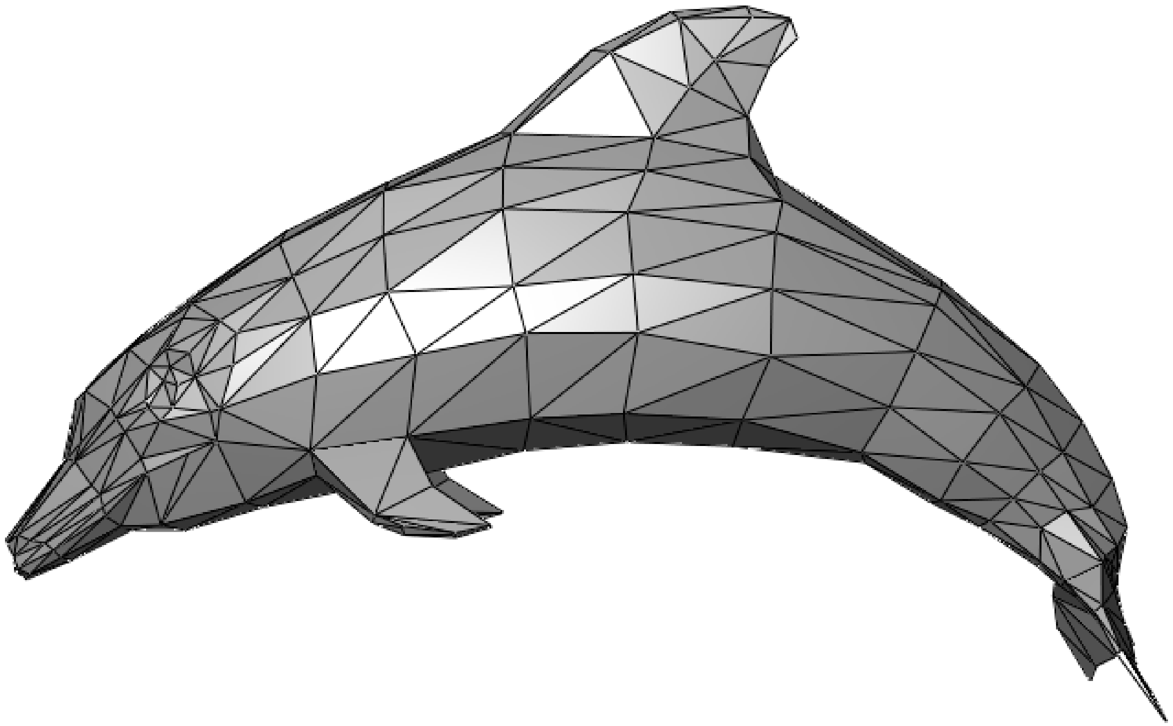
University of Illinois at Urbana-Champaign
{karsch1,vhedau2,daf,dhoiem}@uiuc.edu



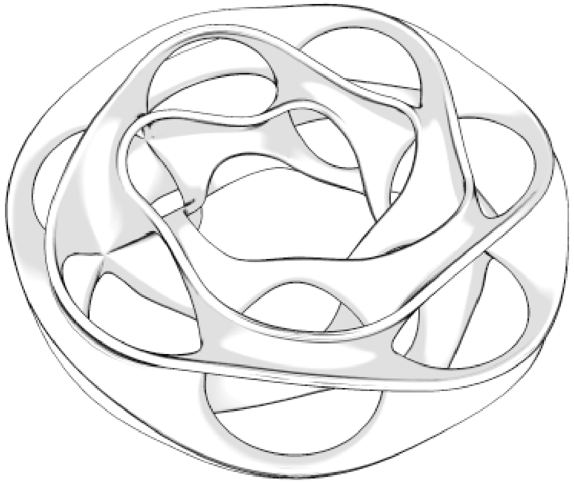
SIGGRAPH ASIA 2011

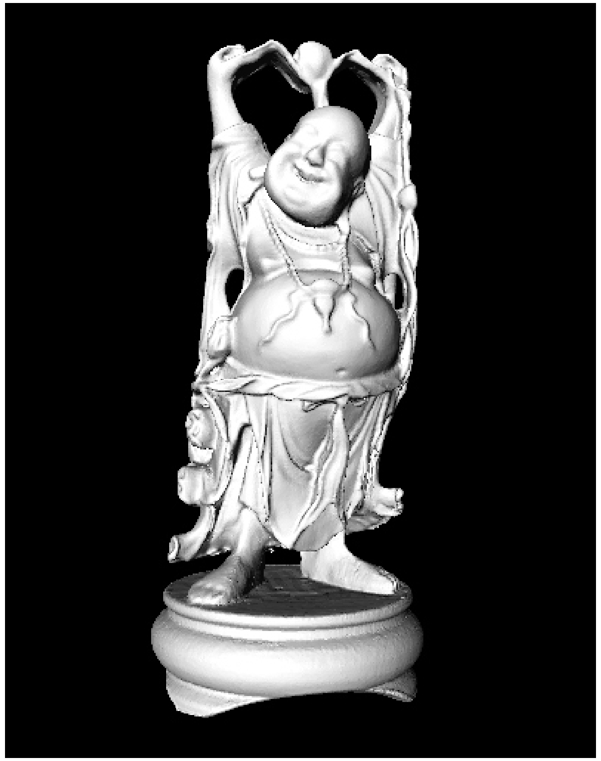
The polygonal mesh

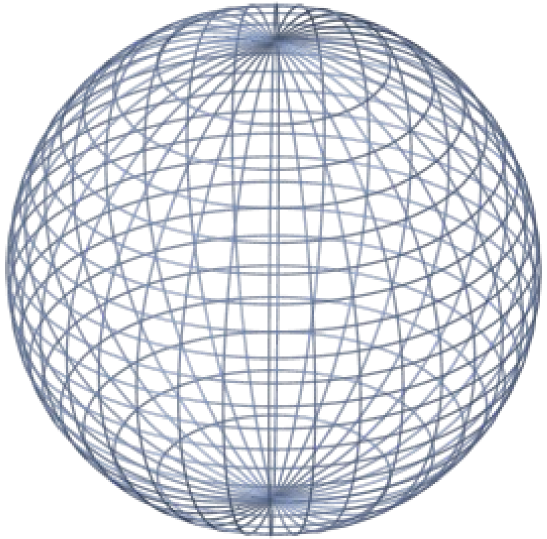
- Discrete representation of a surface
  - Represented by vertices -> edges -> polygons (faces)



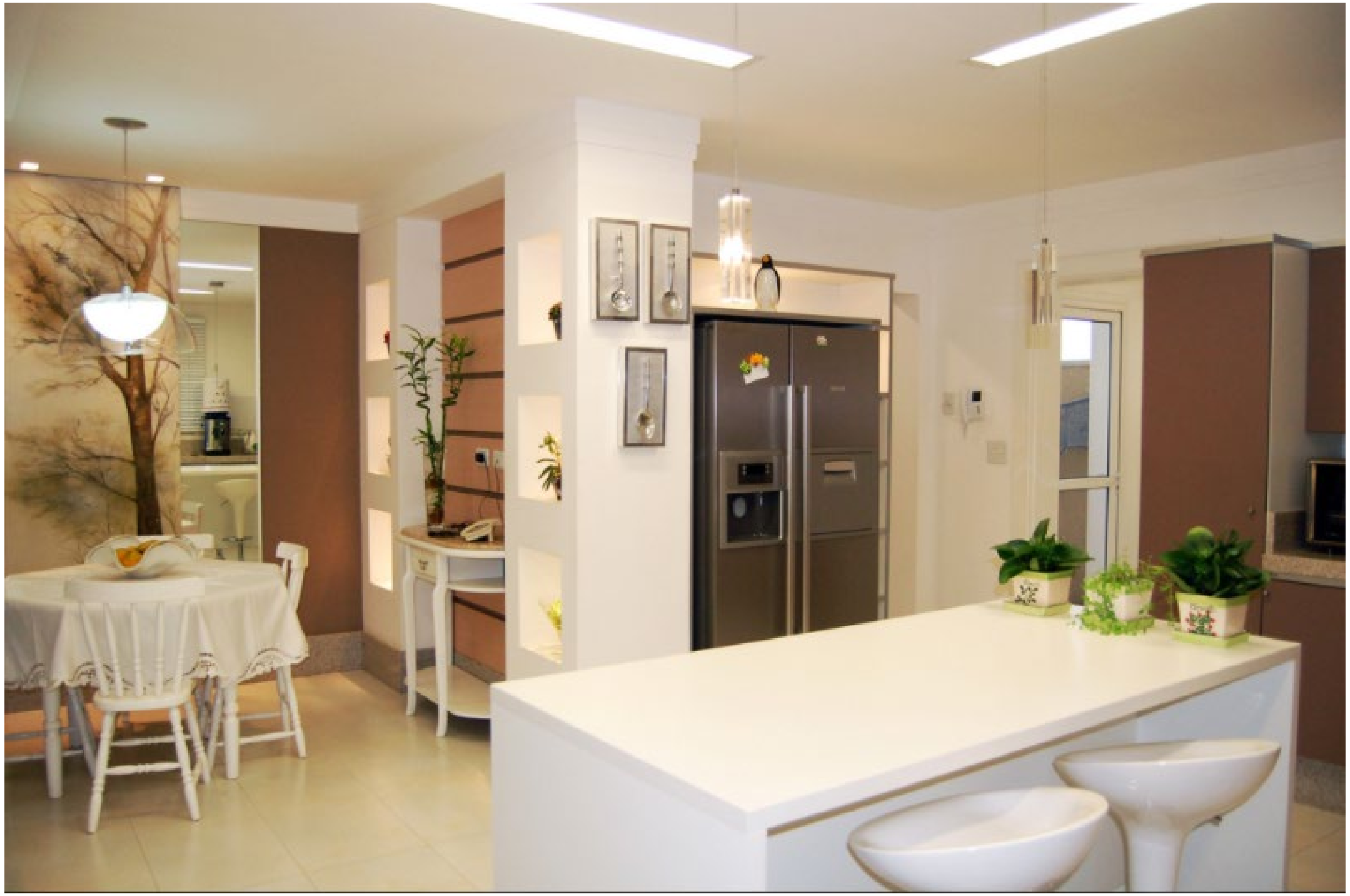
Insert these…



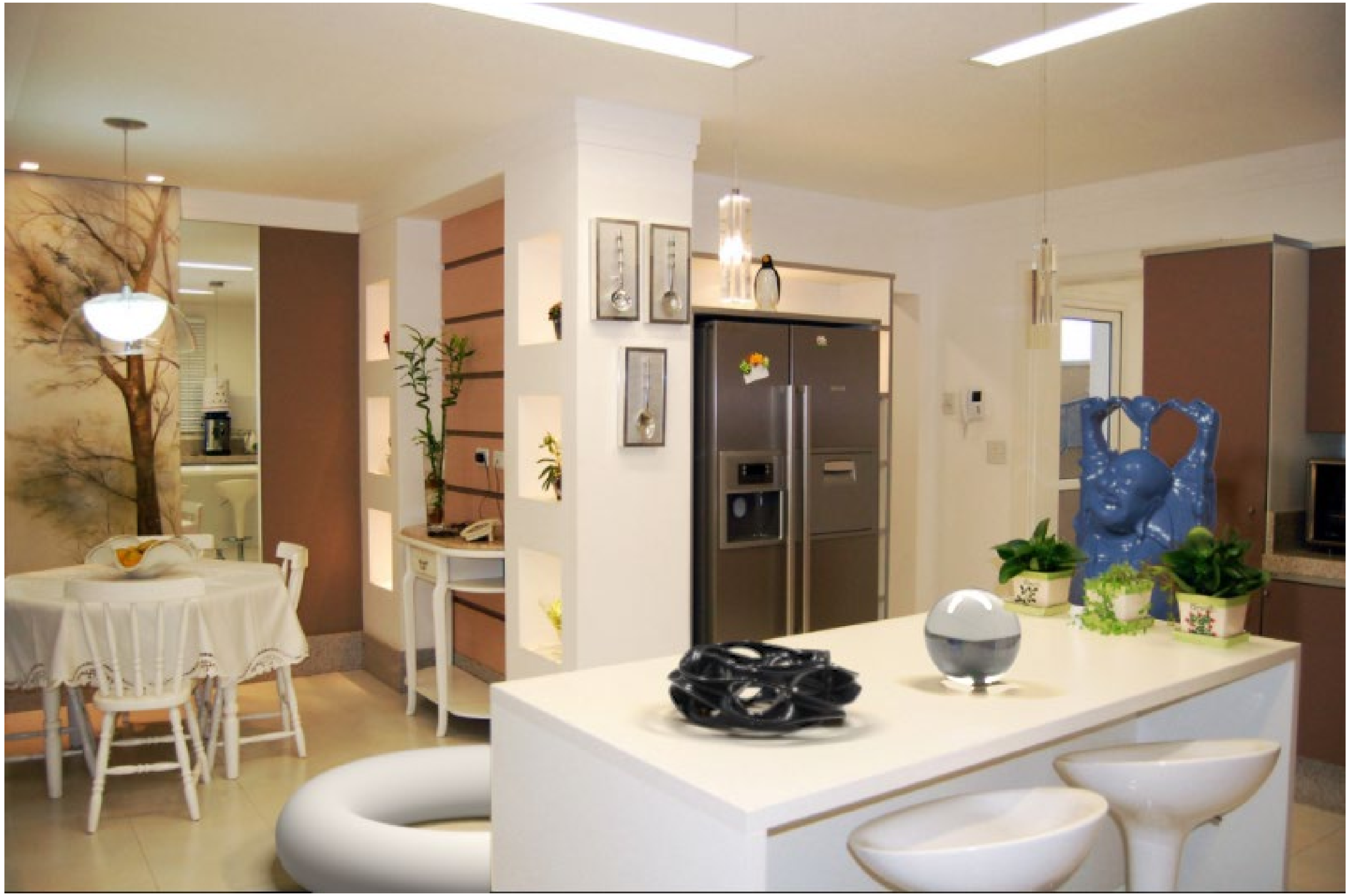




…into this

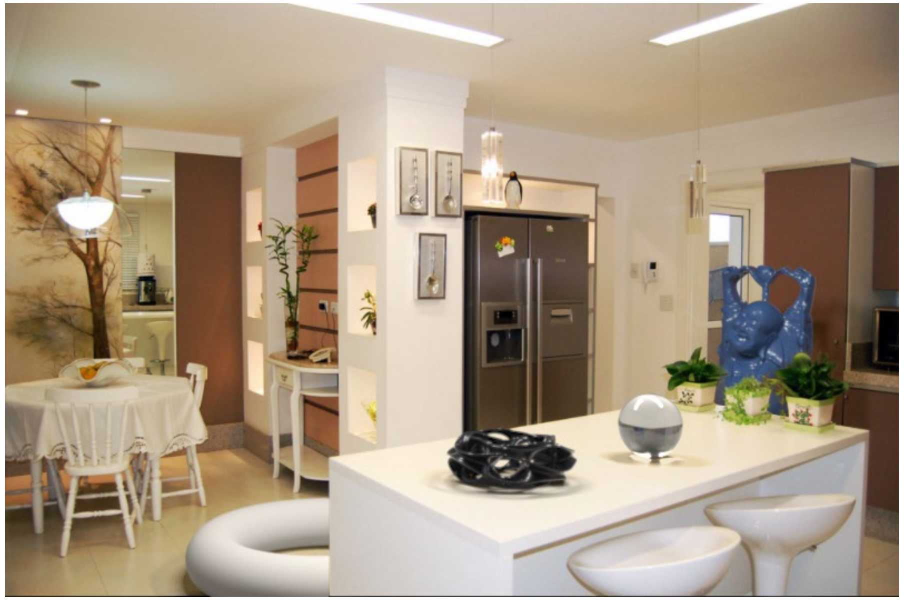


…into this



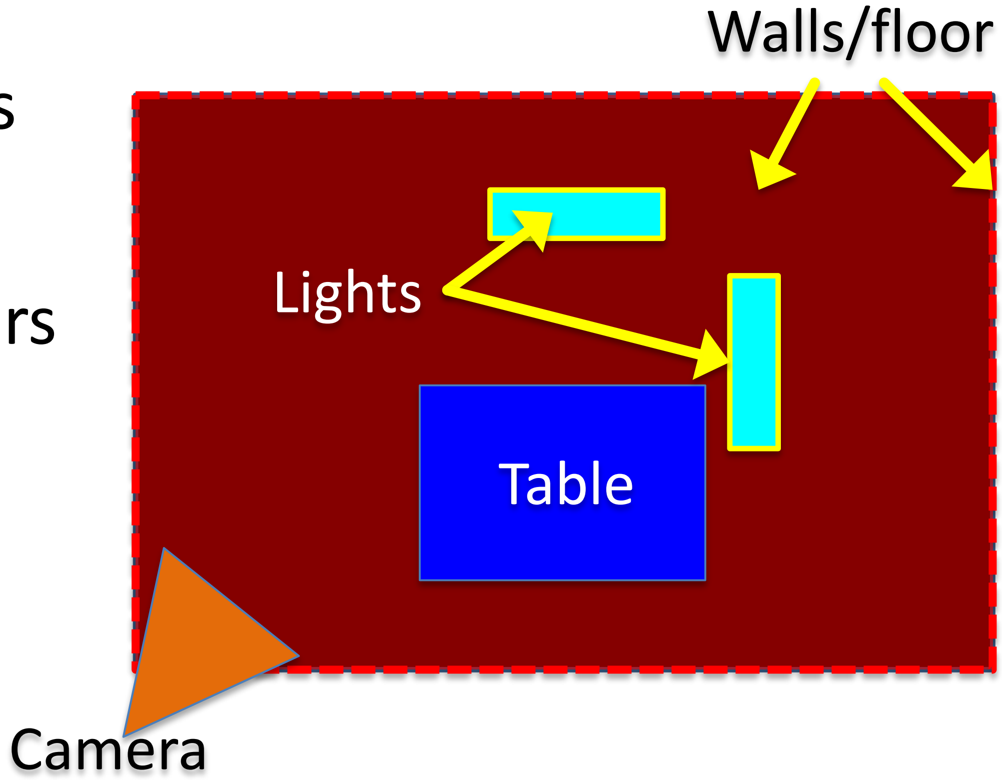
Inserting 3D objects into photographs

- Goal: Realistic insertion using a single LDR photo
- Arbitrary lighting environments
- Intuitive, quick and easy to create content
  - Home planning/redecoration
  - Movies (visual effects)
  - Video games

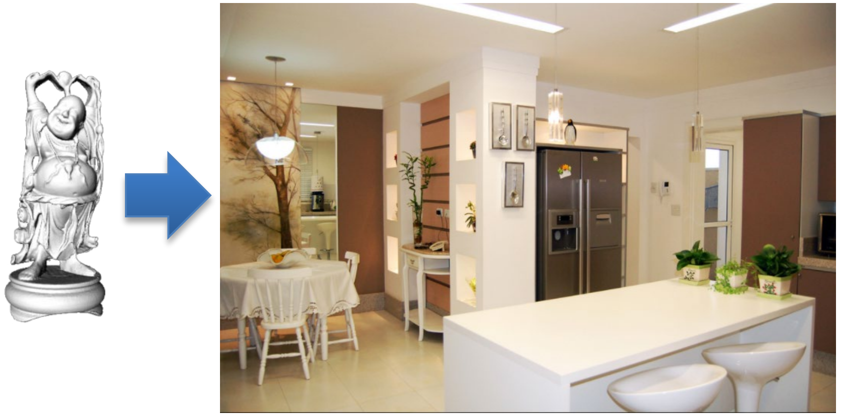


Challenges

- Estimate a physical scene model including:
  - Geometry
  - Surface properties
  - Lighting info
  - Camera parameters



Earlier approaches with scene access



Manual authoring

[Fournier et al. ’93]

Earlier approaches with scene access

Manual authoring

[Fournier et al. ’93]

Light probe, Inverse GI

[Debevec ’98, Yu et al. ‘99]

Earlier approaches without scene access

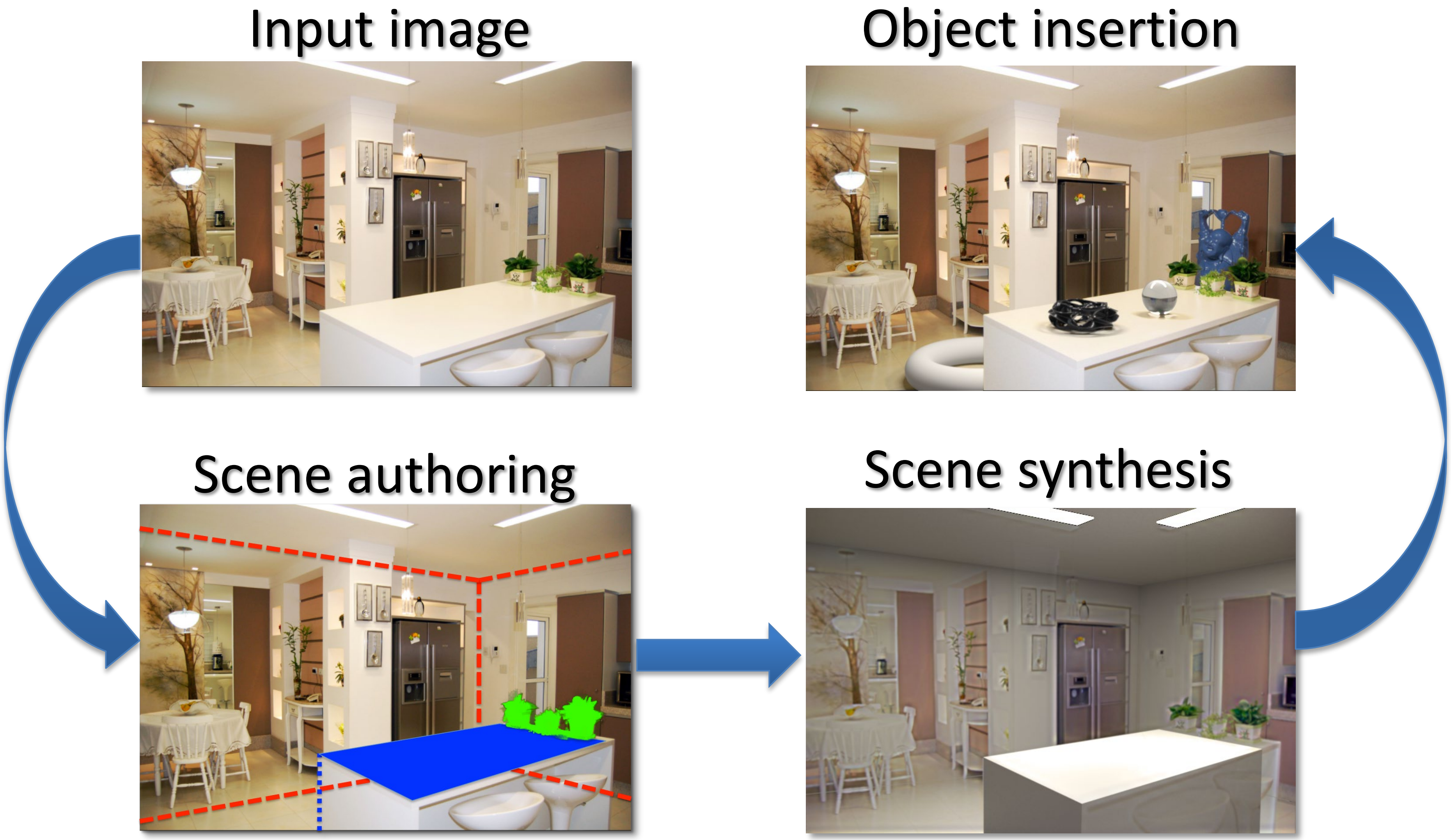
Outdoor illumination

[Lalonde et al. ‘09]

Point source detection

[Wang and Samaras ’03, Lopez-Moreno et al. ‘10]

System overview



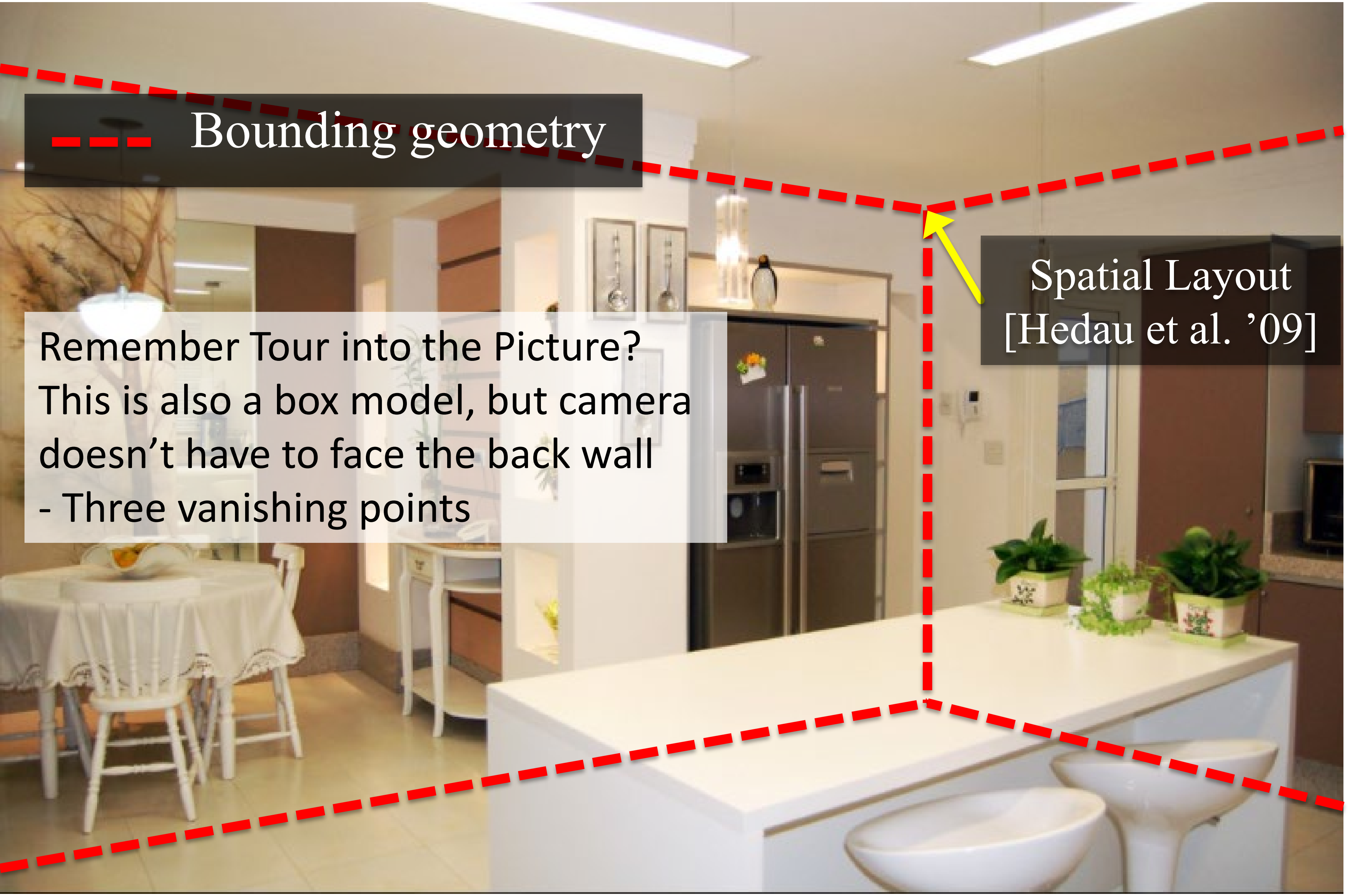
Input image

Object insertion

Scene authoring

Scene synthesis

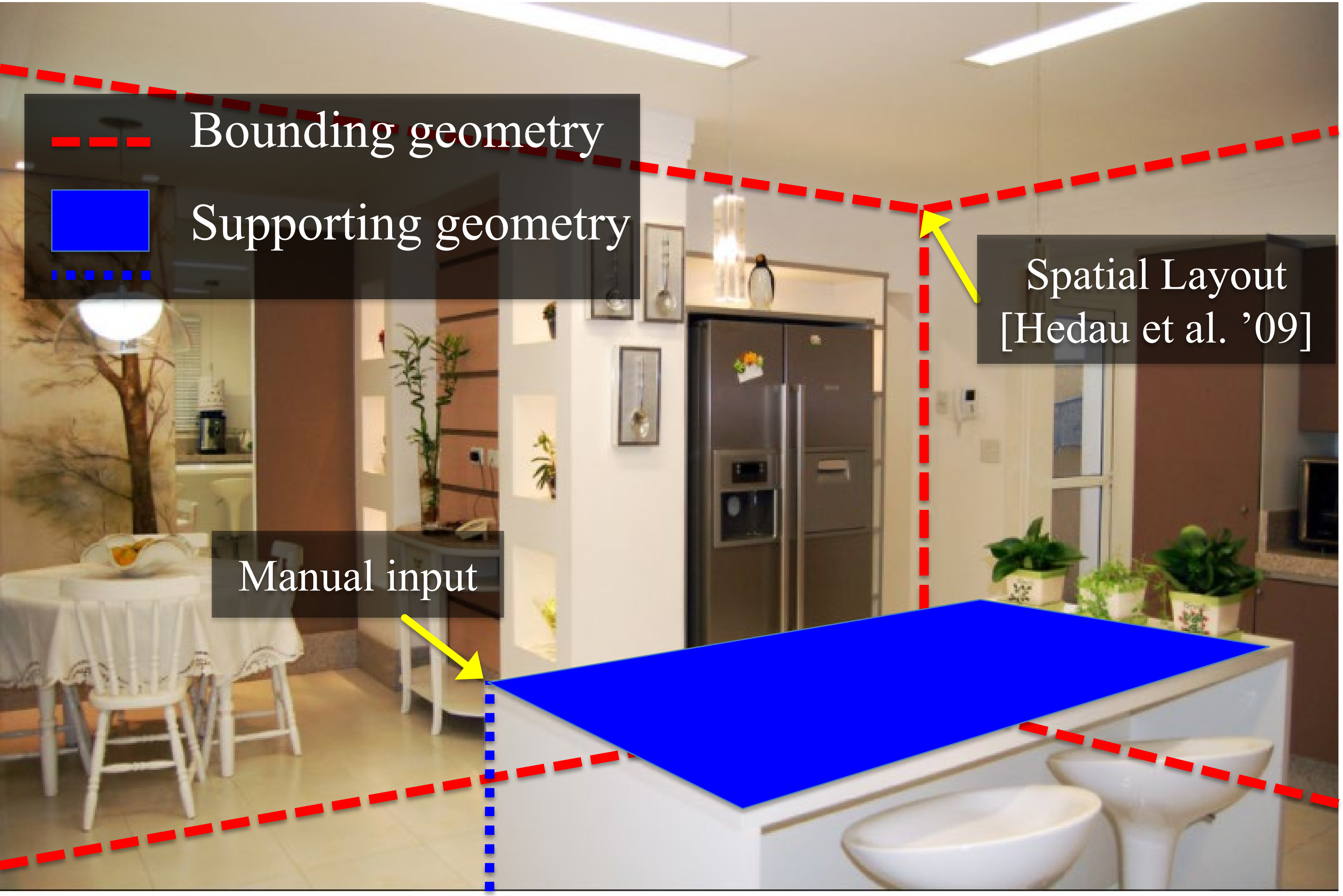
Overview of getting geometry and lighting

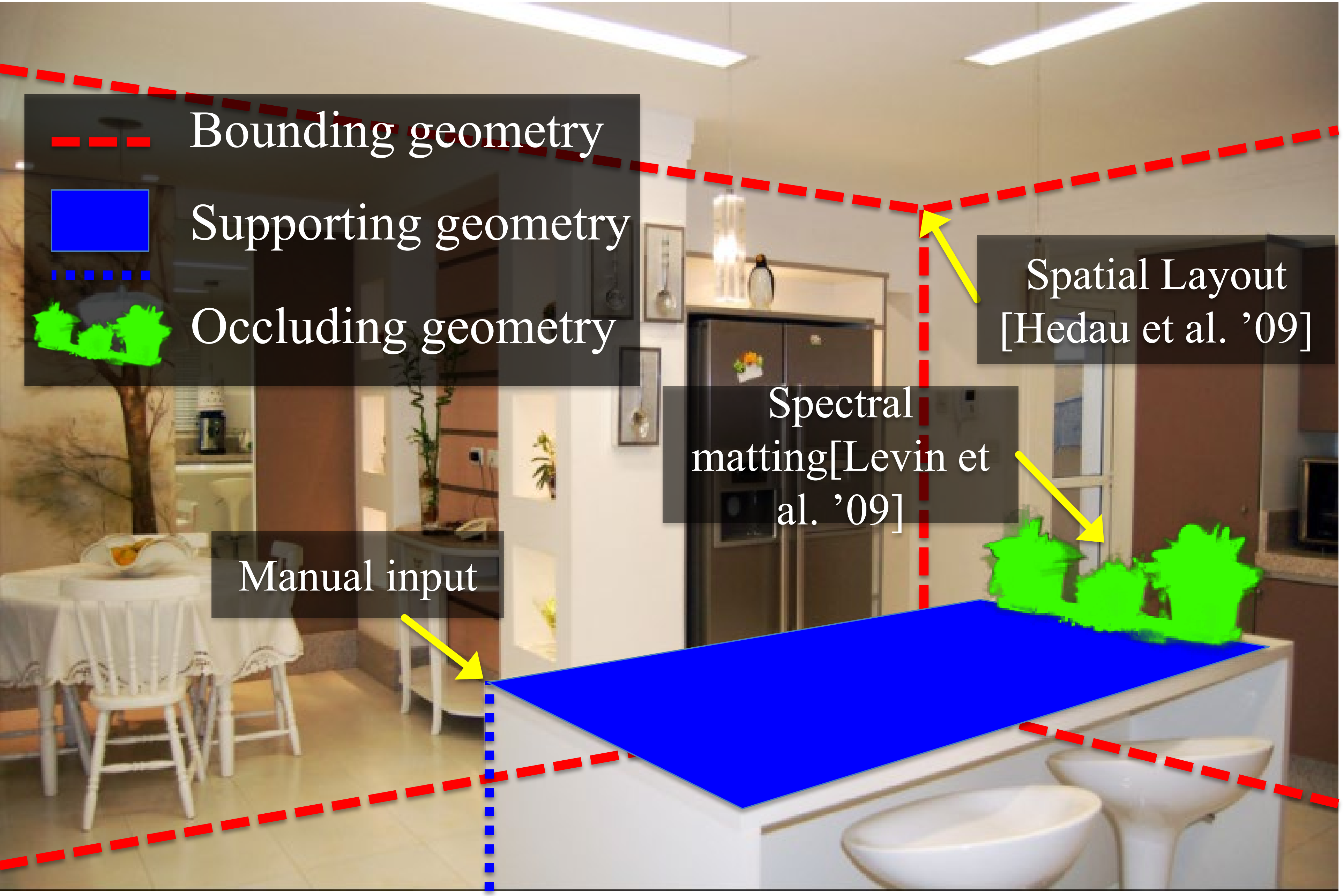


- - - Bounding geometry

Spatial Layout
[Hedau et al. ’09]

Remember Tour into the Picture?
This is also a box model, but camera doesn’t have to face the back wall
- Three vanishing points


Bounding geometry
Supporting geometry
Spatial Layout
[Hedau et al. '09]
Manual input



Bounding geometry

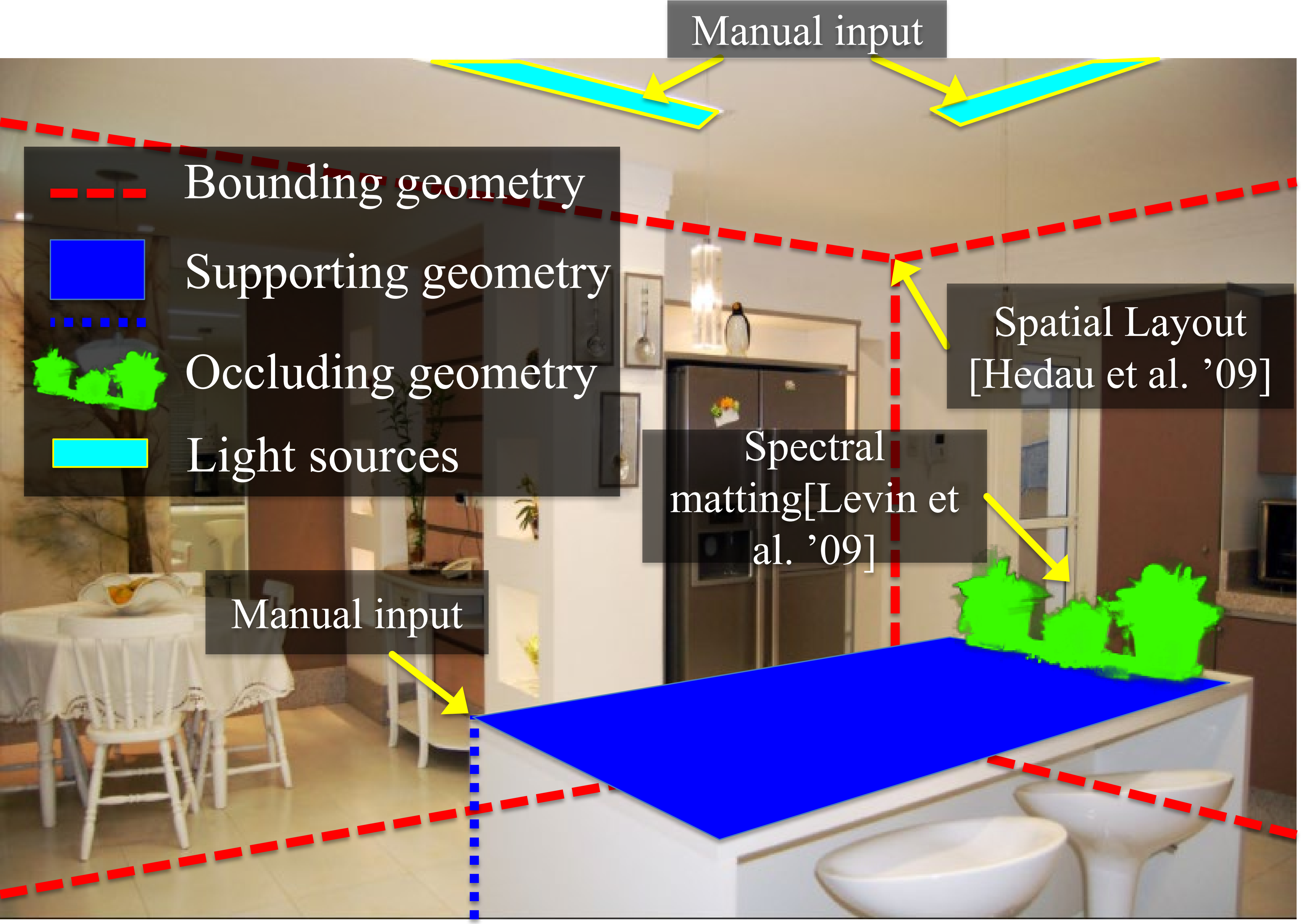
Supporting geometry

Occluding geometry

Spatial Layout [Hedau et al. ’09]

Spectral matting[Levin et al. ’09]

Manual input



Manual input

Bounding geometry

Supporting geometry

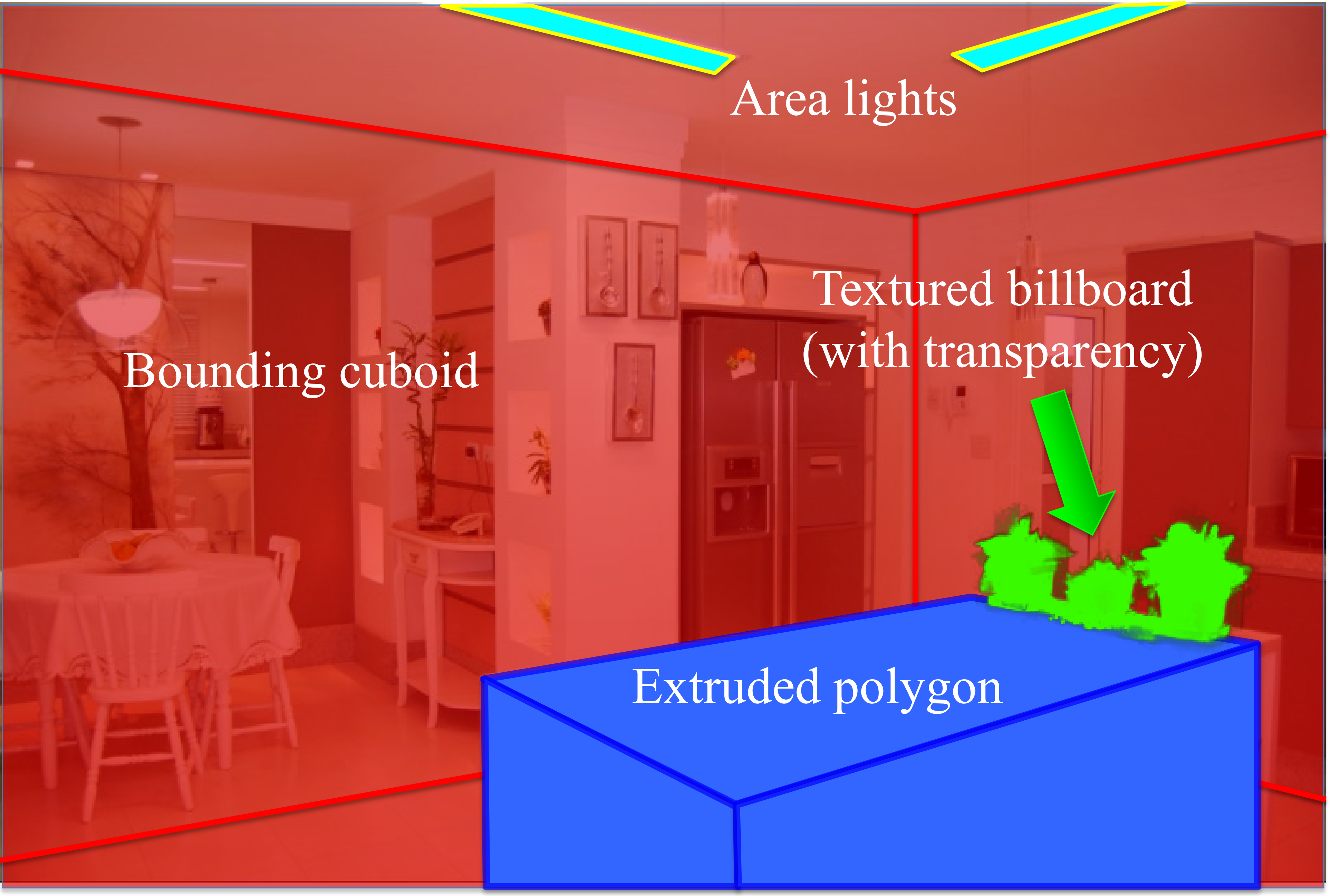
Occluding geometry

Light sources

Spatial Layout [Hedau et al. ’09]

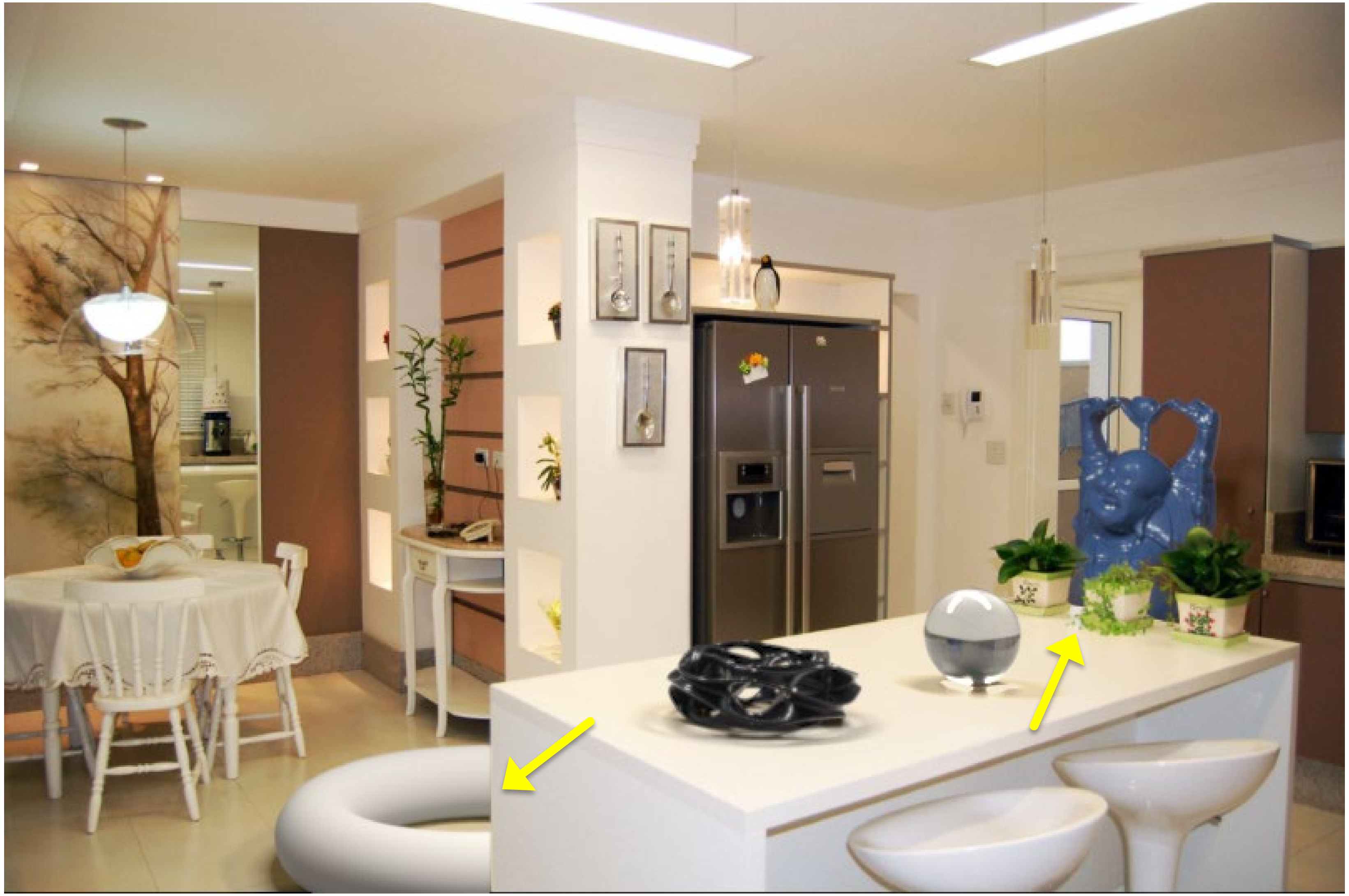
Spectral matting[Levin et al. ’09]

Manual input

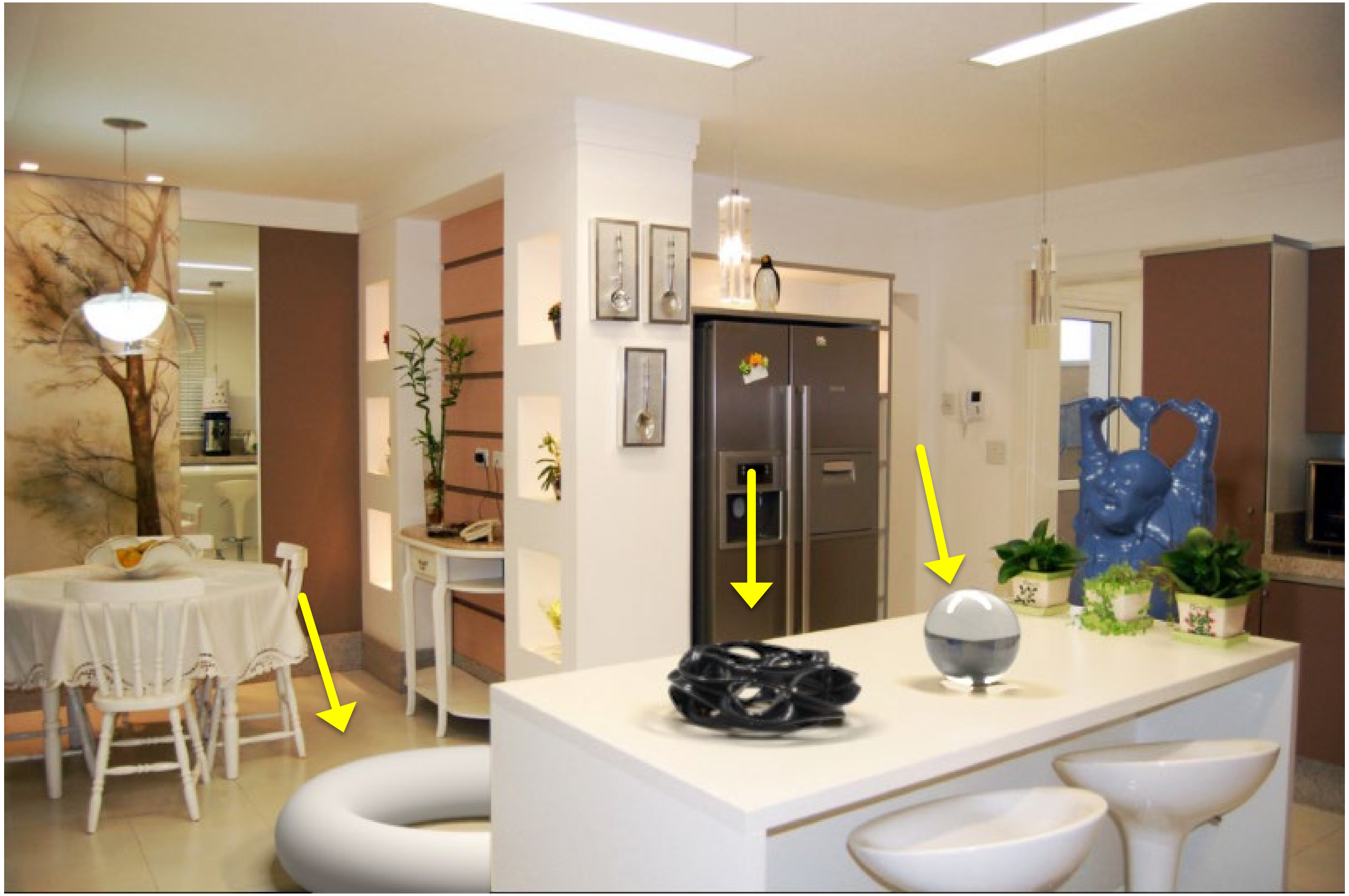

Area lights
Bounding cuboid
Textured billboard
(with transparency)
Extruded polygon

What the spatial layout provides

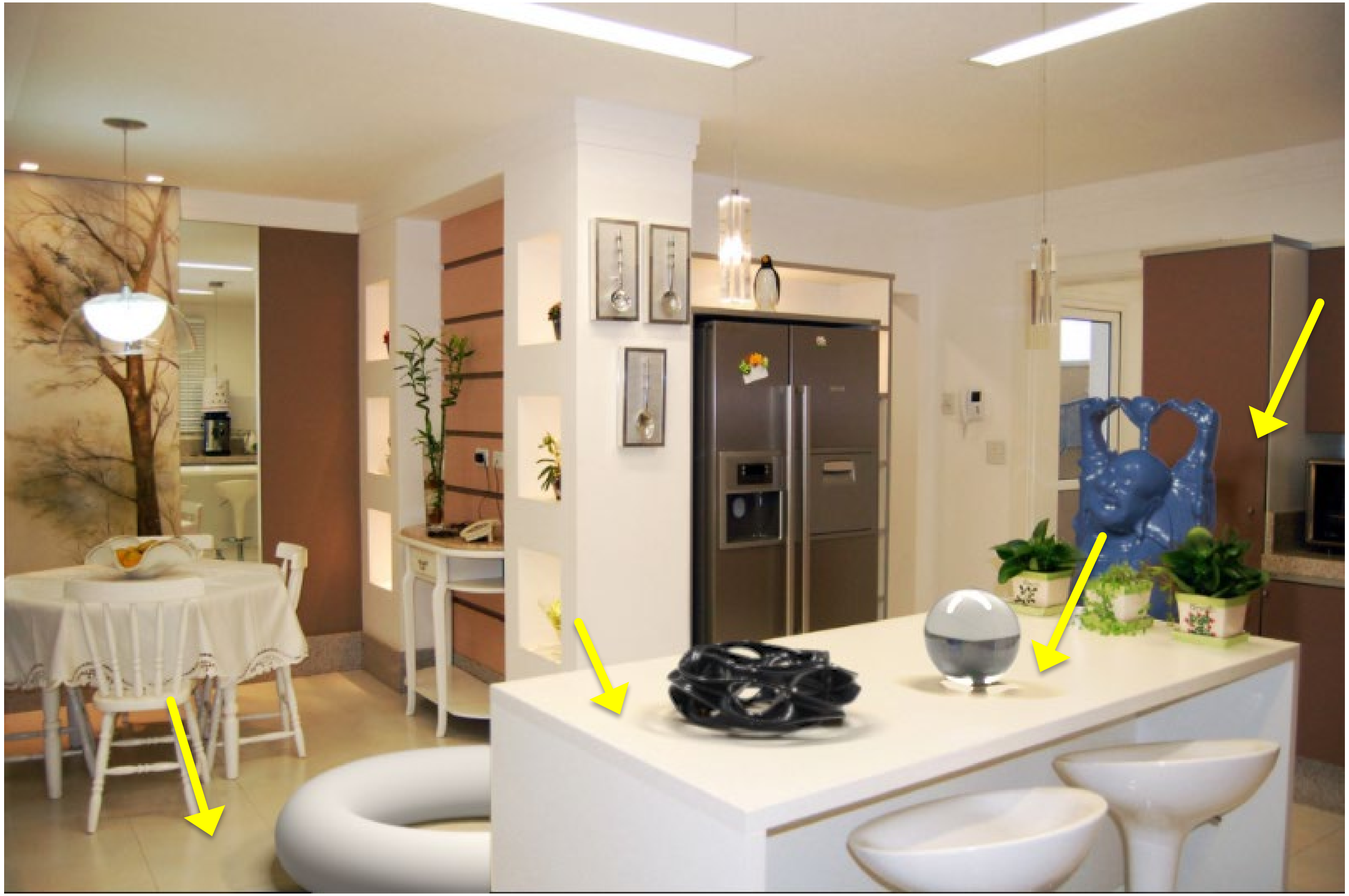
Extruded geometry, billboards enable *occlusion*



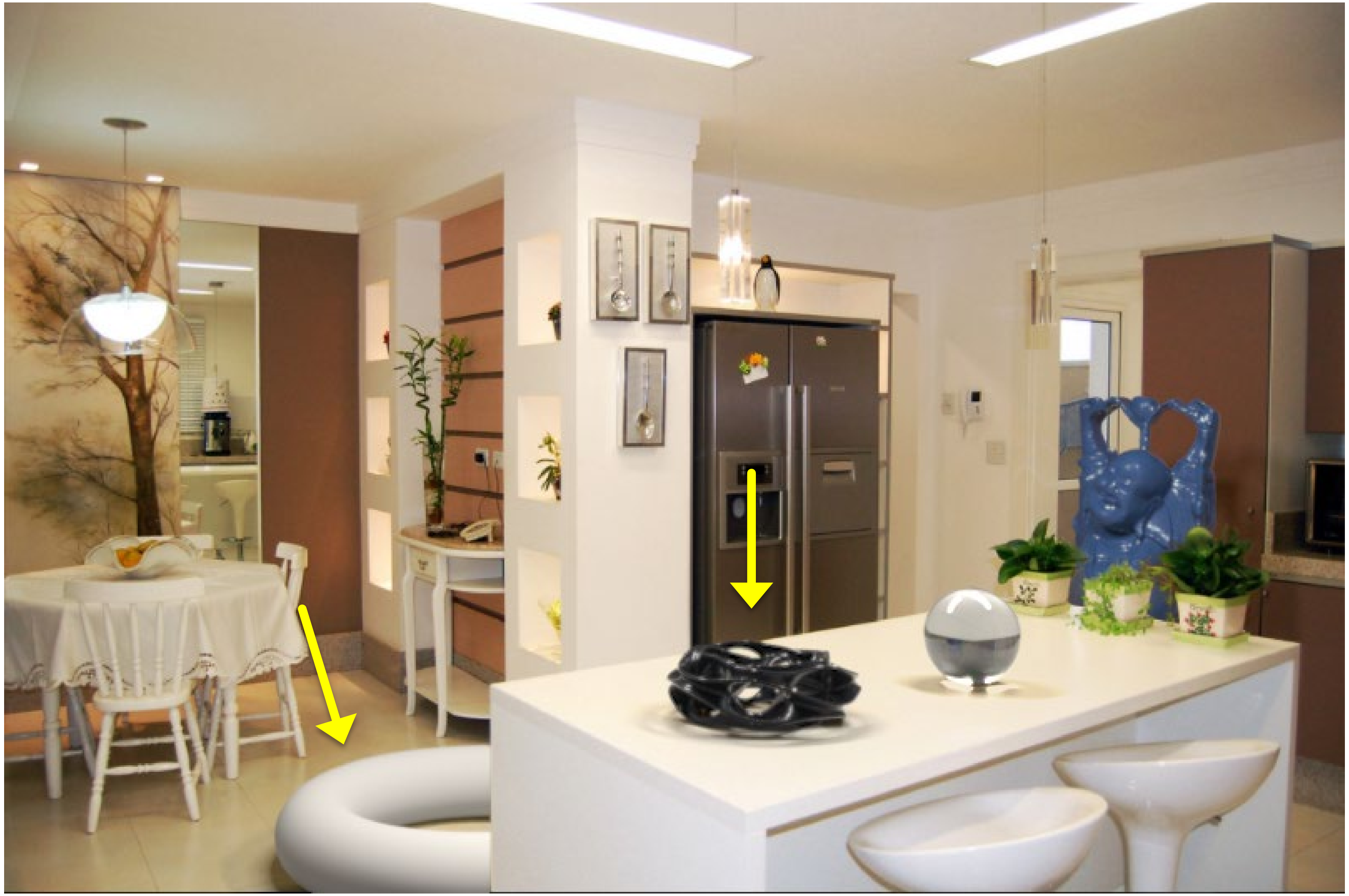
Box, supporting surfaces enable *object placement*

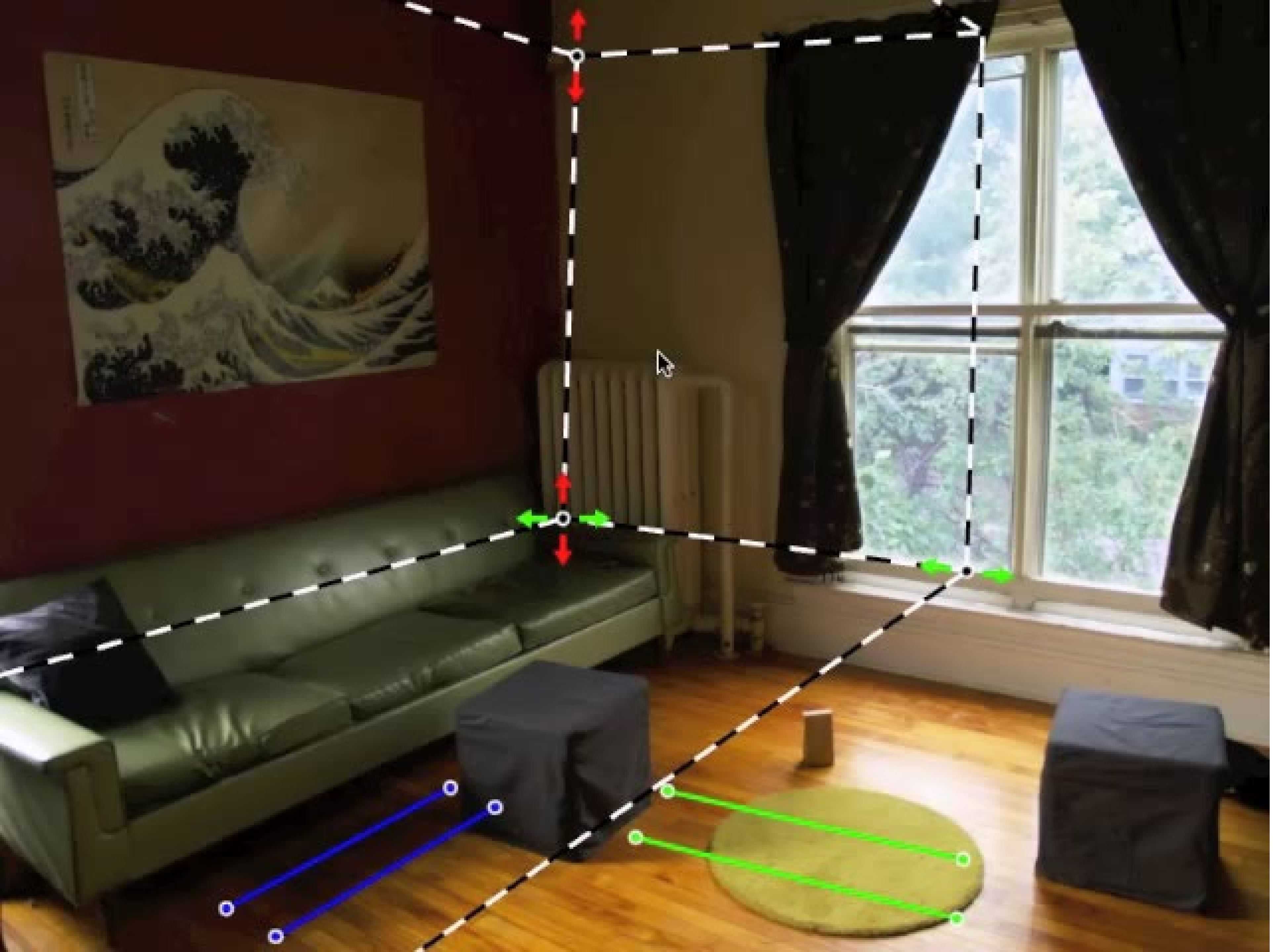


Box, extruded geometry, lighting enables *shadows, inter-reflections, caustics*



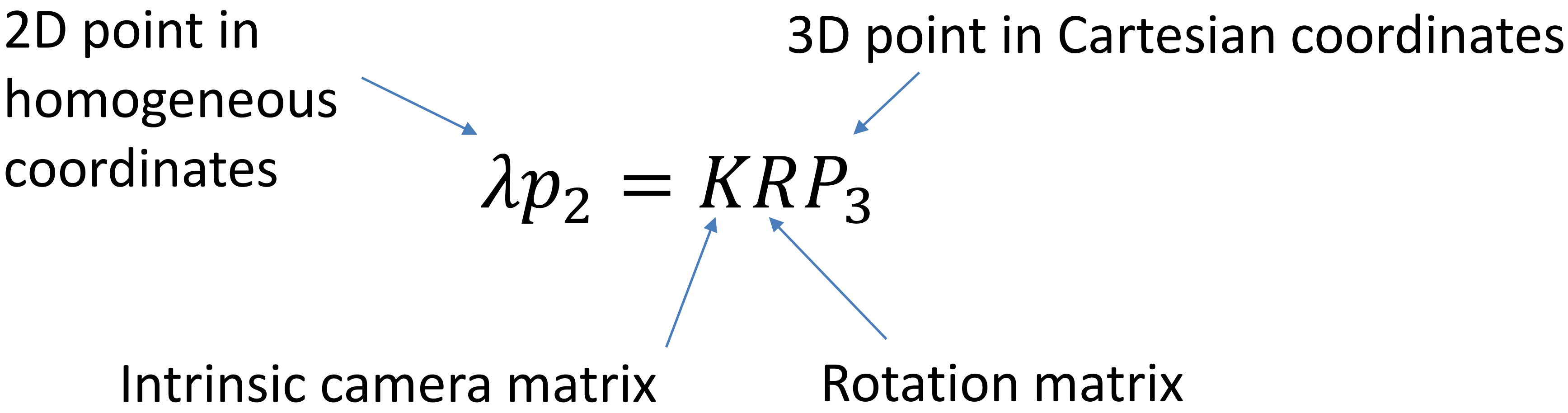
Camera geometry ensures correct *perspective*





Solving for camera viewpoint

Given 3 orthogonal VPs (at least two finite), can compute projection operator



2D point in homogeneous coordinates

3D point in Cartesian coordinates

$$\lambda p_2 = KRP_3$$

Intrinsic camera matrix

Rotation matrix

Solving for camera viewpoint

Given 3 orthogonal VPs (at least two finite), can compute projection operator: intrinsic matrix

$$K = \begin{bmatrix} f & 0 & u_0 \\ 0 & f & v_0 \\ 0 & 0 & 1 \end{bmatrix} K^{-1} = \begin{bmatrix} 1/f & 0 & -u_0/f \\ 0 & 1/f & -v_0/f \\ 0 & 0 & 1 \end{bmatrix}$$

$$e_i = (1,0,0)^T, e_j = (0,1,0)^T, e_k = (0,0,1)^T$$
$$v_i = KRe_i, v_j = KRe_j, v_k = KRe_k$$
$$(KR)^{-1}v_i = e^i, (KR)^{-1}v_j = e^j, (KR)^{-1}v_k = e^k$$

$$e_i^T e_j = e_j^T e_k = e_i^T e_k = 0$$
$$v_i^T K^{-T} R R^{-1} K^{-1} v_j = v_j^T K^{-T} R R^{-1} K^{-1} v_k = v_i^T K^{-T} R R^{-1} K^{-1} v_k = 0$$
$$v_i^T K^{-T} K^{-1} v_j = v_j^T K^{-T} K^{-1} v_k = v_i^T K^{-T} K^{-1} v_k = 0$$

Solving for camera viewpoint

Given 3 orthogonal VPs (at least two finite), can compute projection operator

$$R = \begin{bmatrix} R_{1c} & R_{2c} & R_{3c} \end{bmatrix}$$

$$\lambda v_i = KRe_i \qquad e_i = [1, 0, 0]^T$$

$$R_{ic} = \lambda K^{-1} v_i$$

Projecting to image space

Given K, R, and a position in 3D, we can find its corresponding 2D image location:

$$\lambda p_2 = KRP_3$$

What about the reverse?

Given K, R, and a 2D position on the image, what do we know about its 3D location?

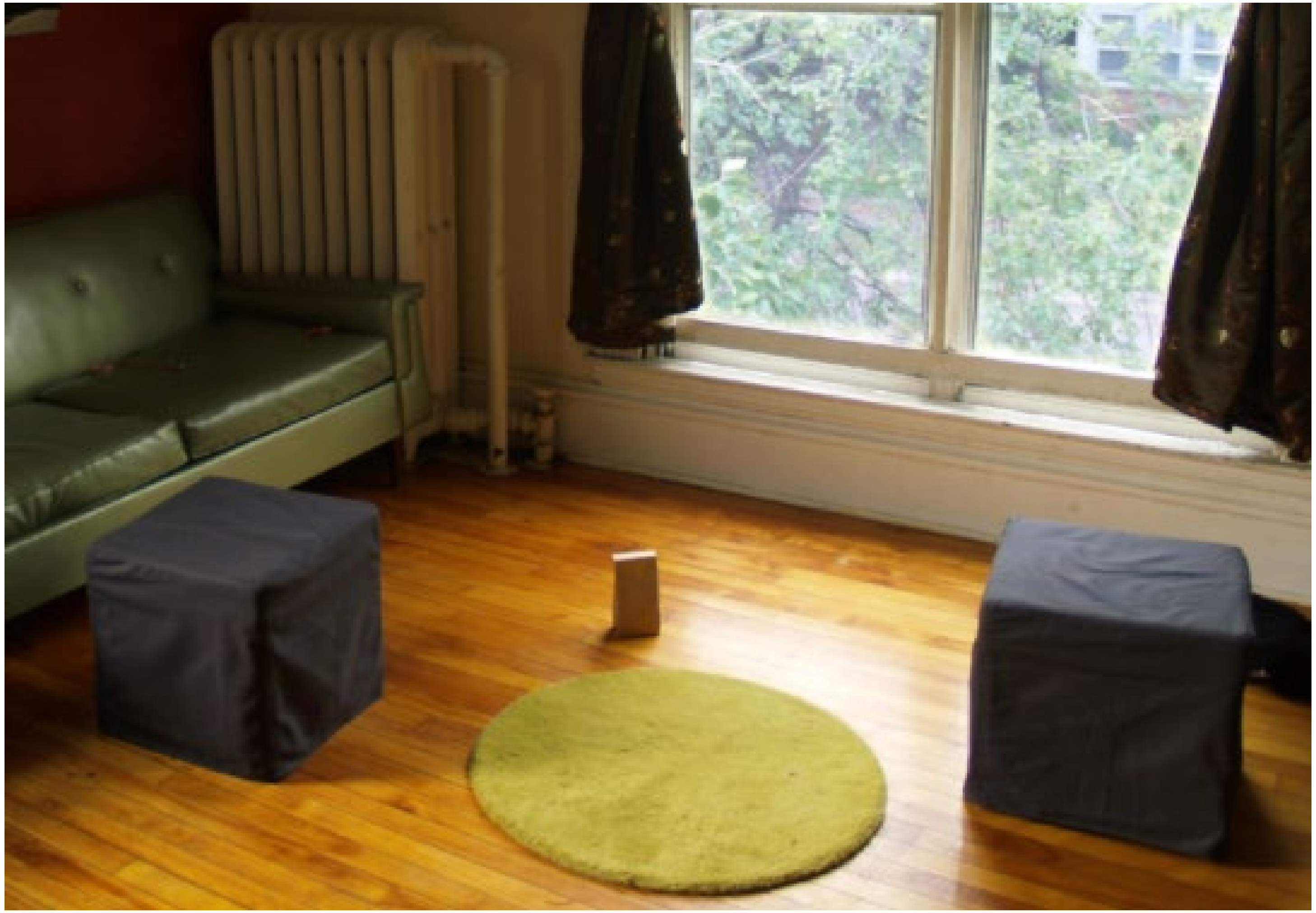
What about the reverse?

Given K, R, and a 2D position on the image, what do we know about its 3D location?

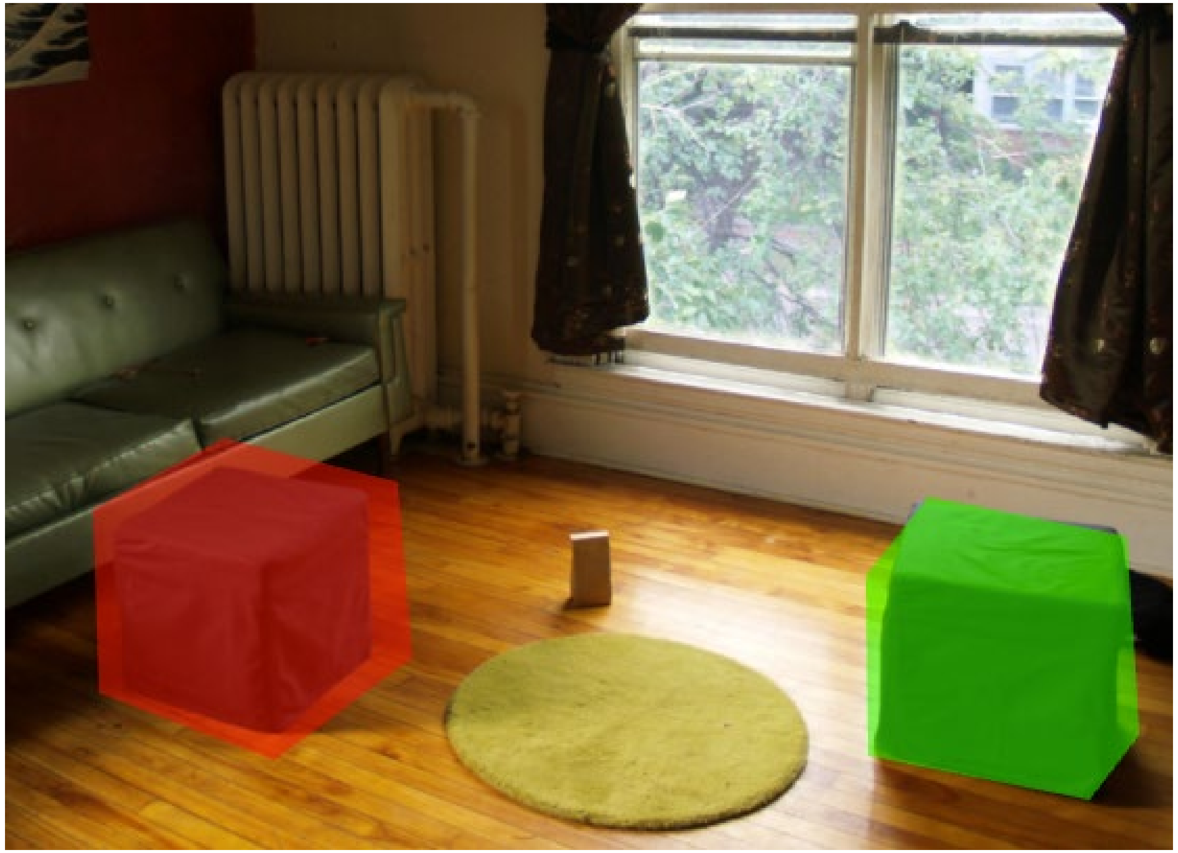
$$(KR)^{-1}p_2 = \lambda P_3$$

- Implies a line along which the 3D point lies
- Points on known surfaces can be localized

Modeling occlusions

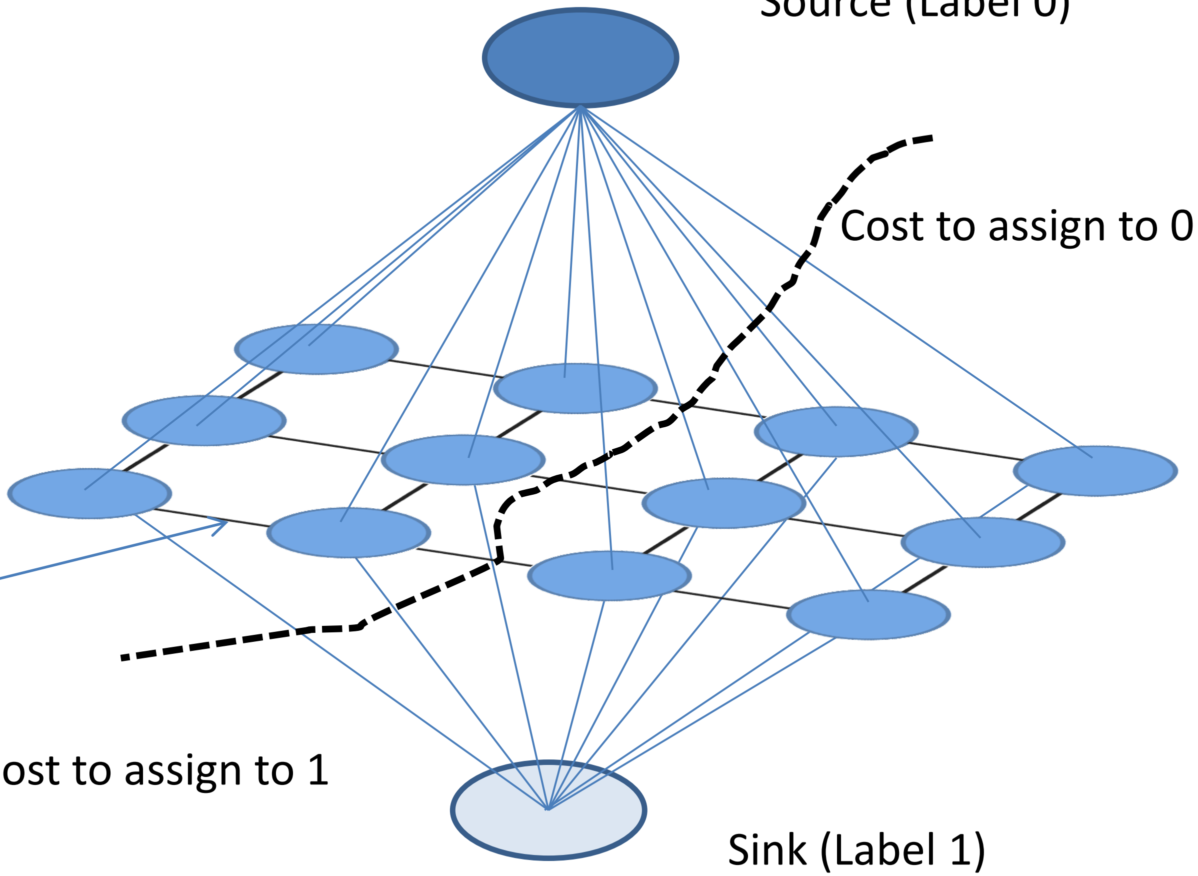


User-defined boundary



- Tedious/inaccurate
- How can we make this better?

Segmentation with graph cuts



Source (Label 0)

Cost to assign to 0

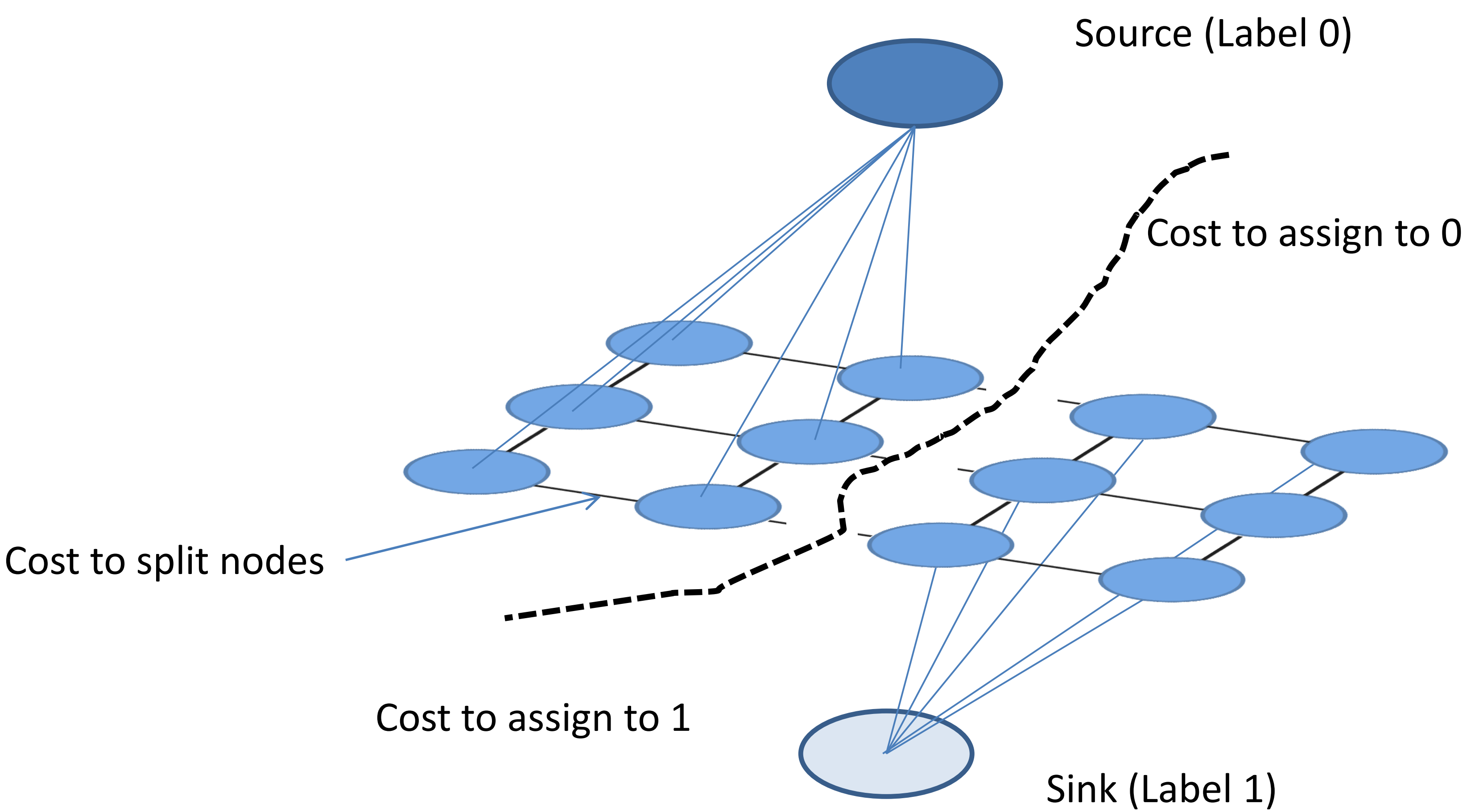
Cost to split nodes

Cost to assign to 1

Sink (Label 1)

$$Energy(\mathbf{y};\theta,data)=\sum_{i}\psi_1(y_i;\theta,data)\ \sum_{i,j\in edges}\psi_2(y_i,y_j;\theta,data)$$

Segmentation with graph cuts



Source (Label 0)

Cost to assign to 0

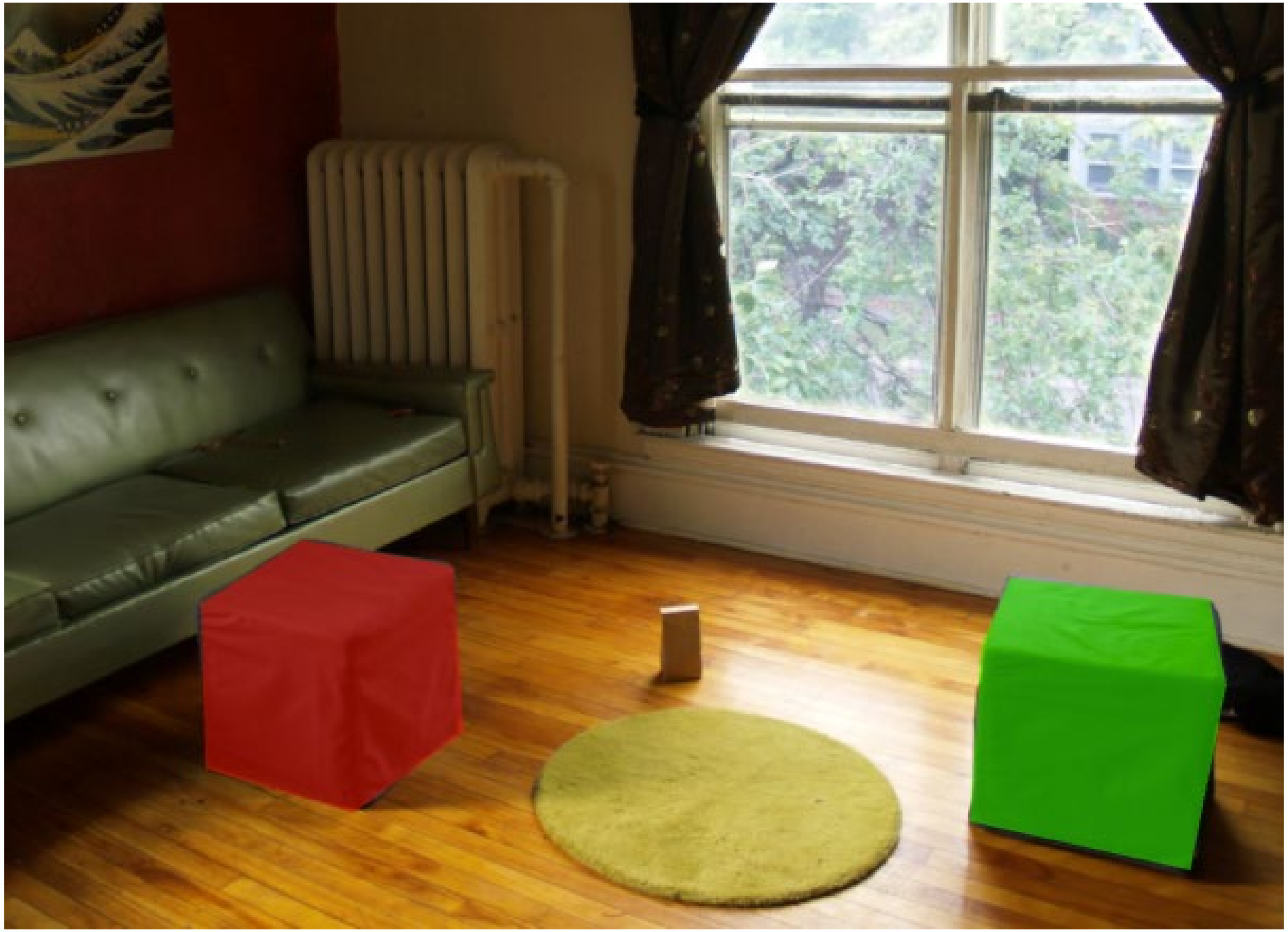
Cost to split nodes

Cost to assign to 1

Sink (Label 1)

$$Energy(\mathbf{y};\theta,data)=\sum_{i}\psi_1(y_i;\theta,data)\ \sum_{i,j\in edges}\psi_2(y_i,y_j;\theta,data)$$

Refined segmentation



Spectral Matting

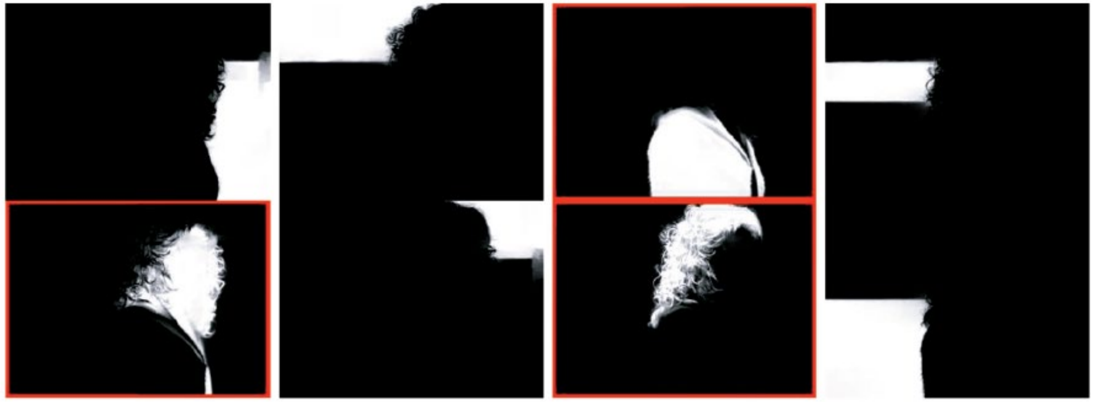
Spectral Matting

- Create NxN matrix describing neighboring pixel similarity (Laplacian matrix, L)
- Extract “smallest” eigenvectors of L
- Soft segmentation defined by linear combination of eigenvectors
  - Scribbles provide constraints to assign to foreground

[“Spectral Matting” Levin Rav-Acha Lischinski 2008]

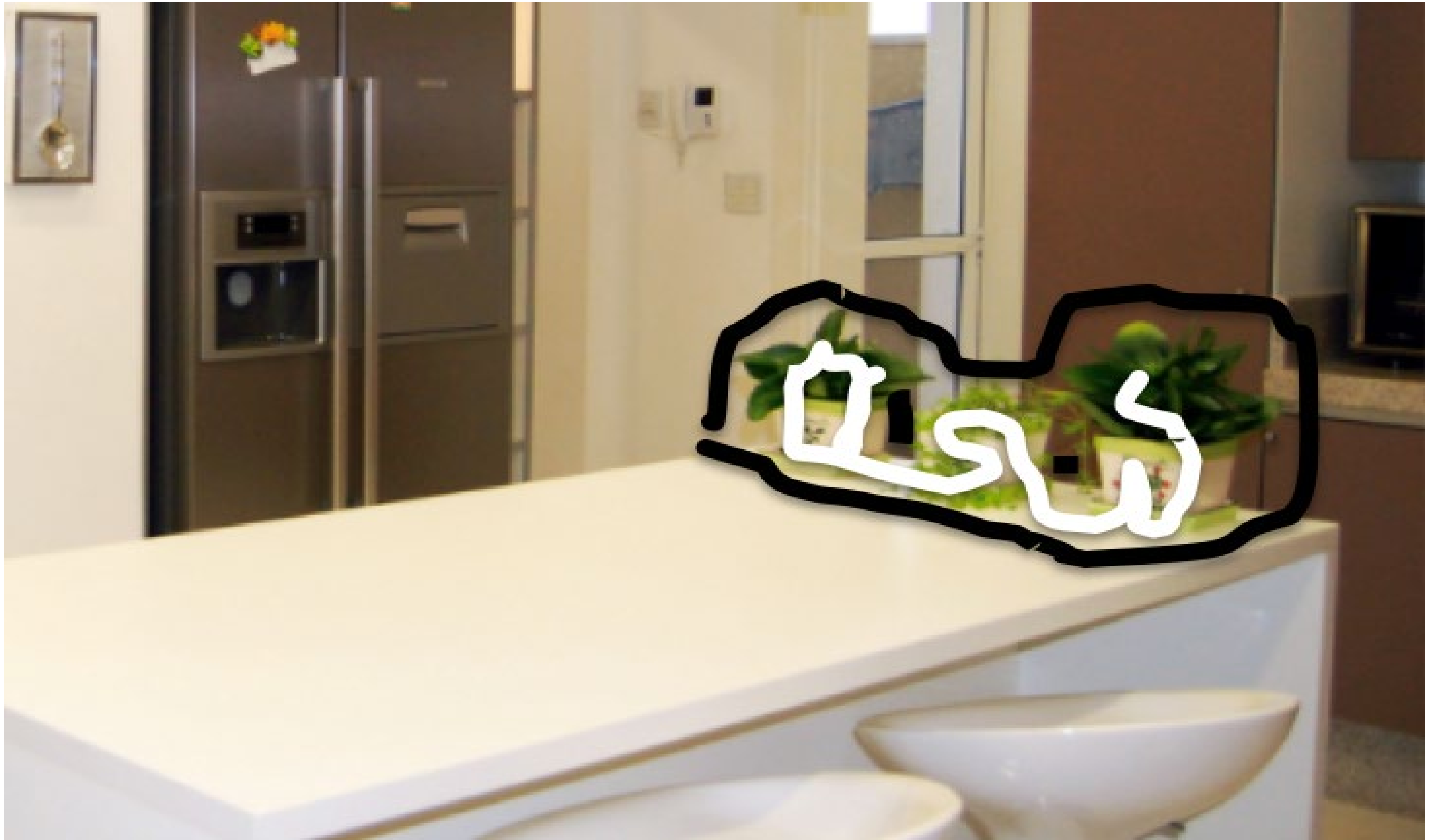
Spectral Matting

image

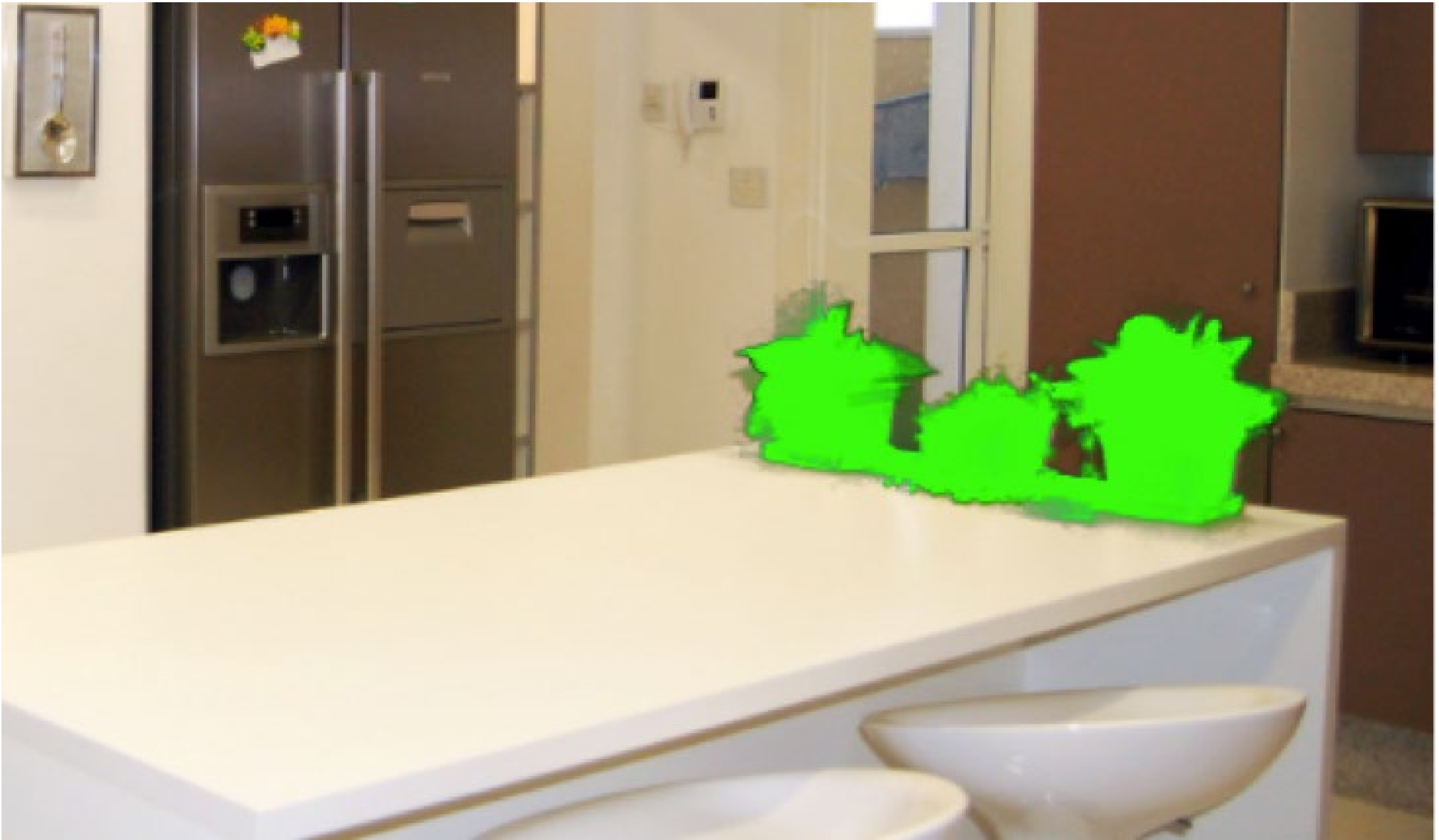


spectral components

Spectral matting

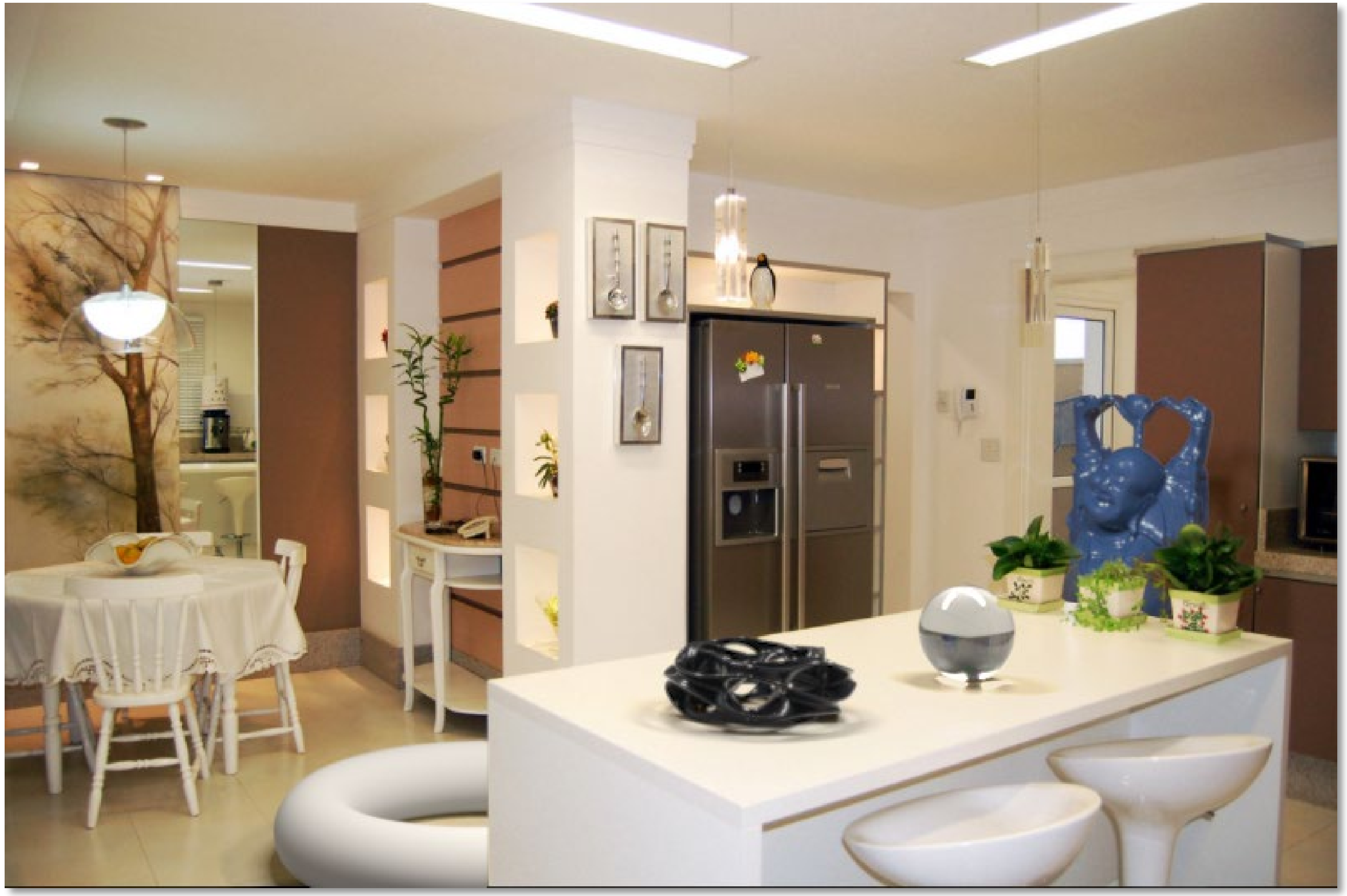


Spectral matting

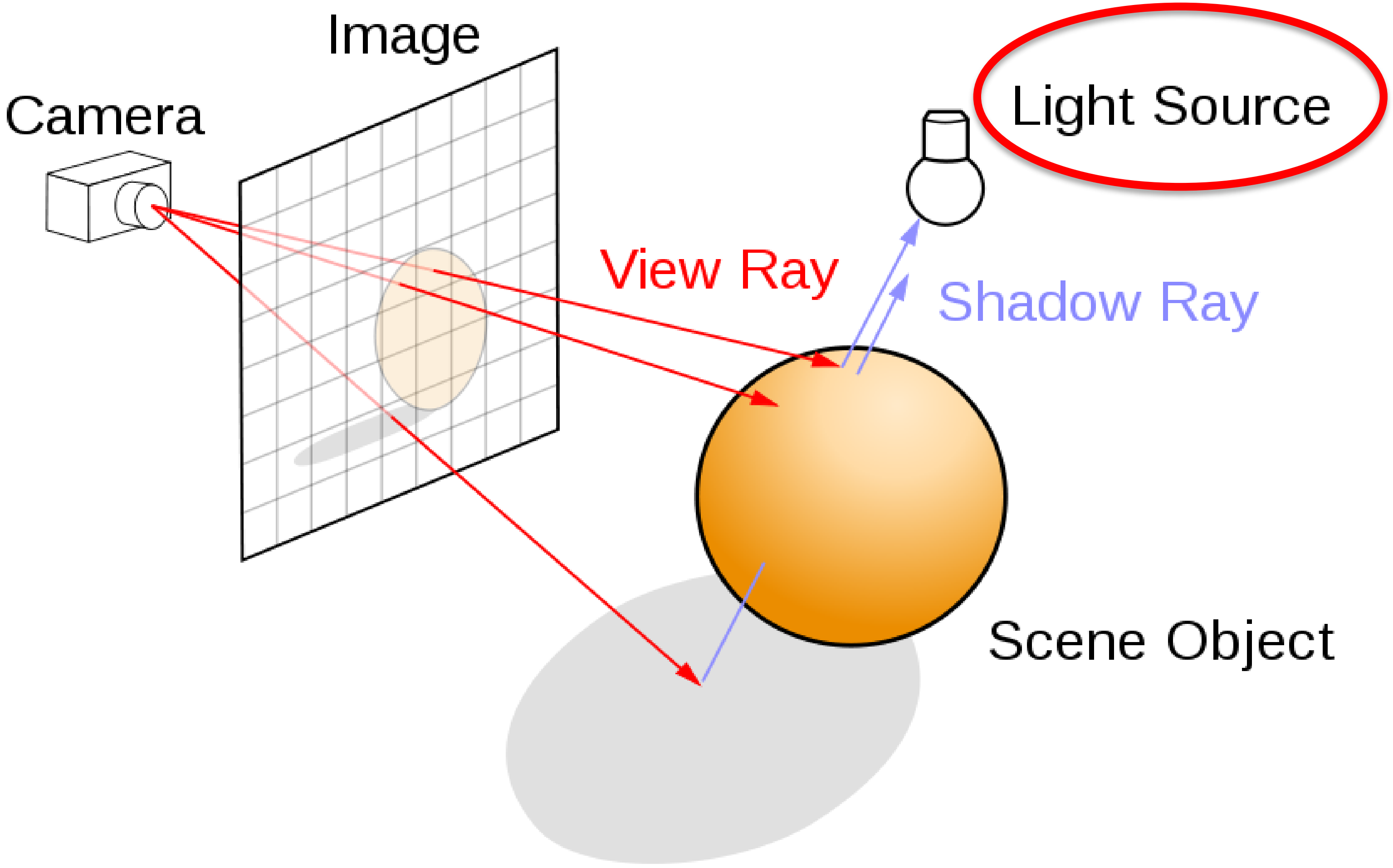


Segmentations as “billboards”

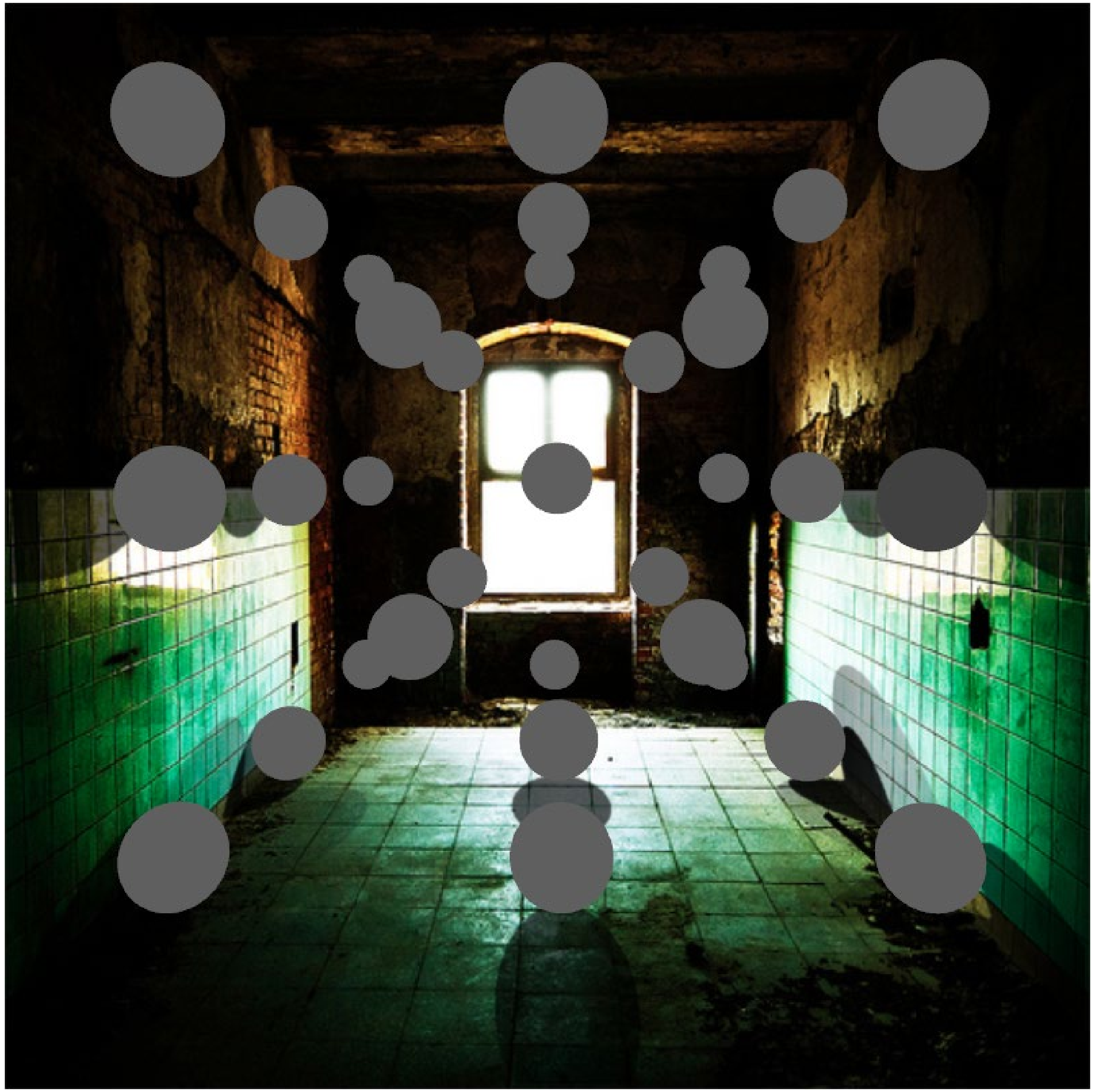
Segmentations as “billboards”



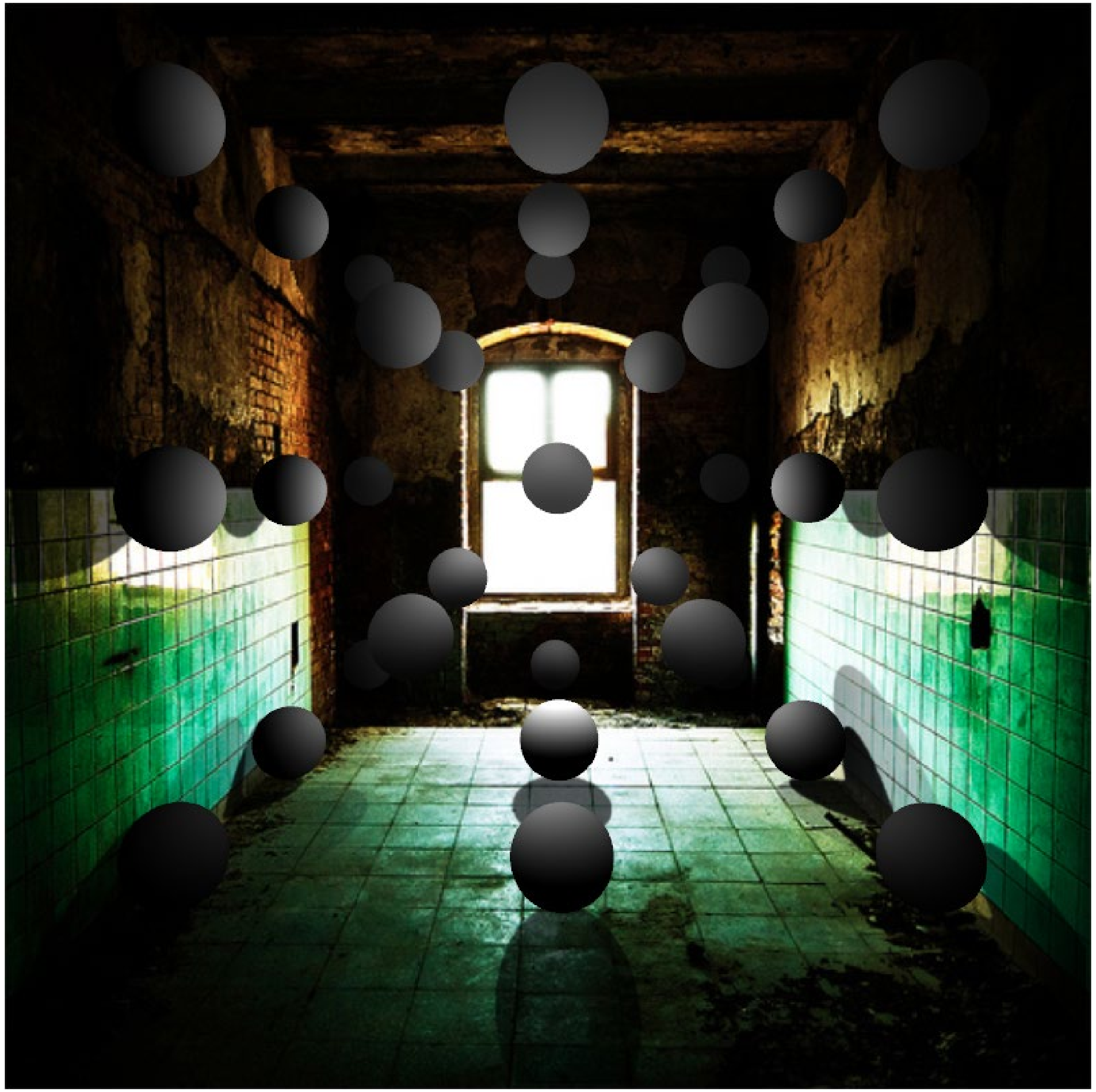
Rendering via ray tracing



Insertion without relighting



…with relighting

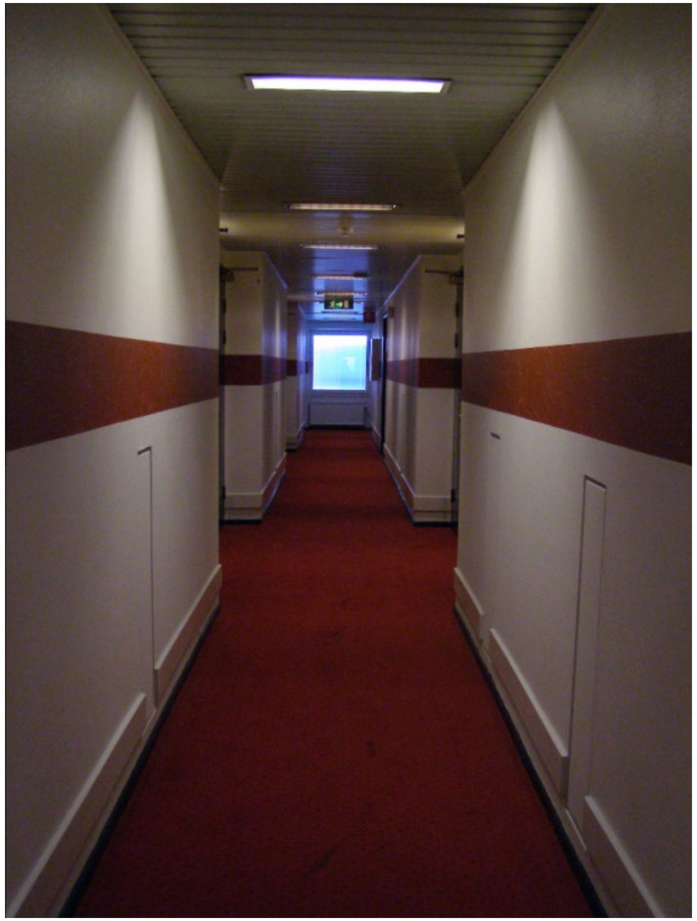


Estimating light

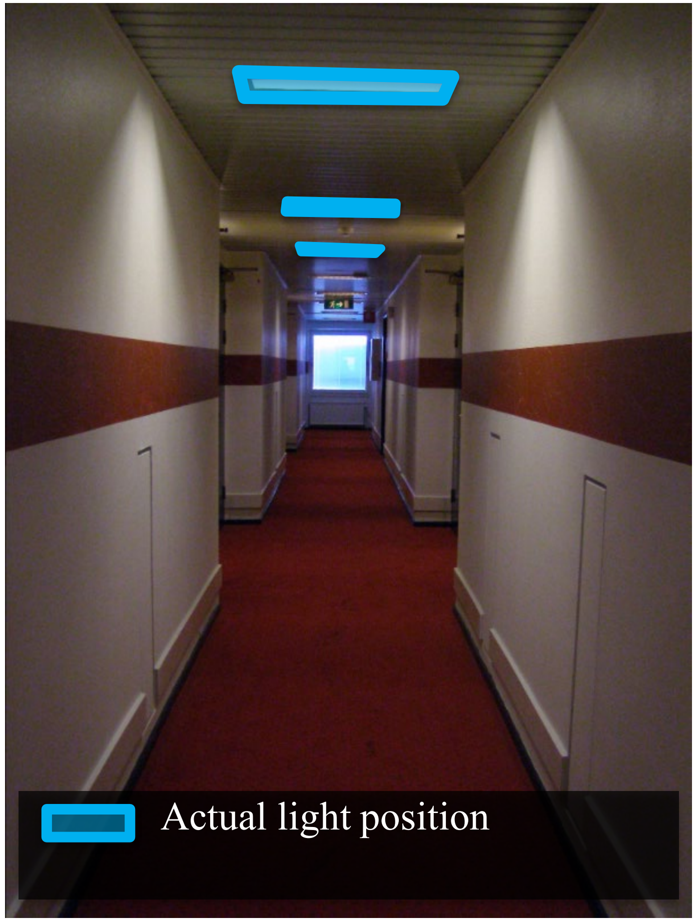
- Hypothesize physical light sources in the scene
  - Physical → CG representations of light sources found in the real world (area lights, etc)

- Visible sources in image marked by user
  - Refined to best match geometry and materials
- User annotates light shafts; direction vector
  - Shafts automatically matted and refined

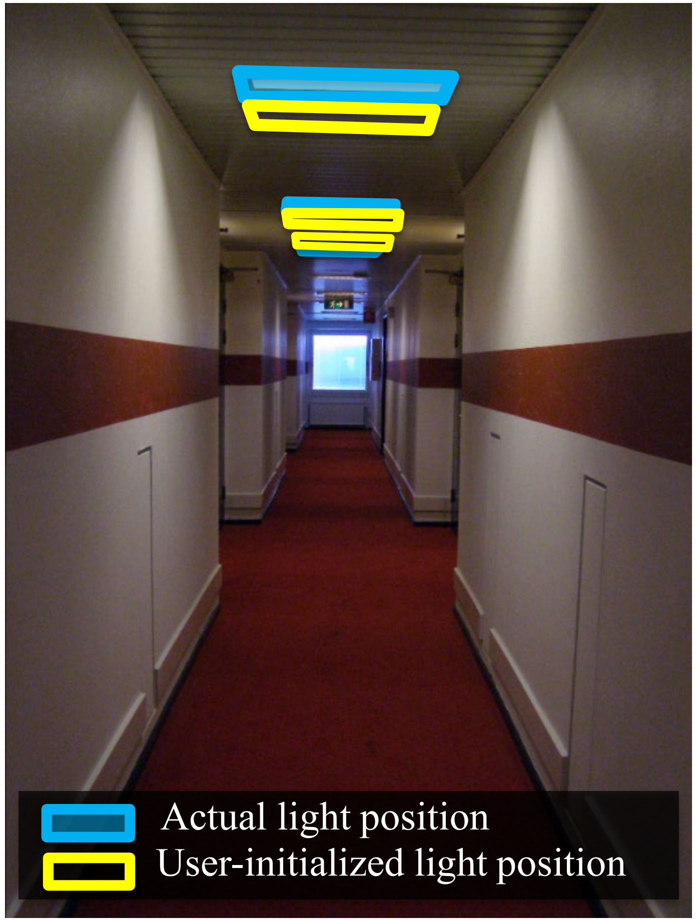
Lighting estimation



Lighting estimation

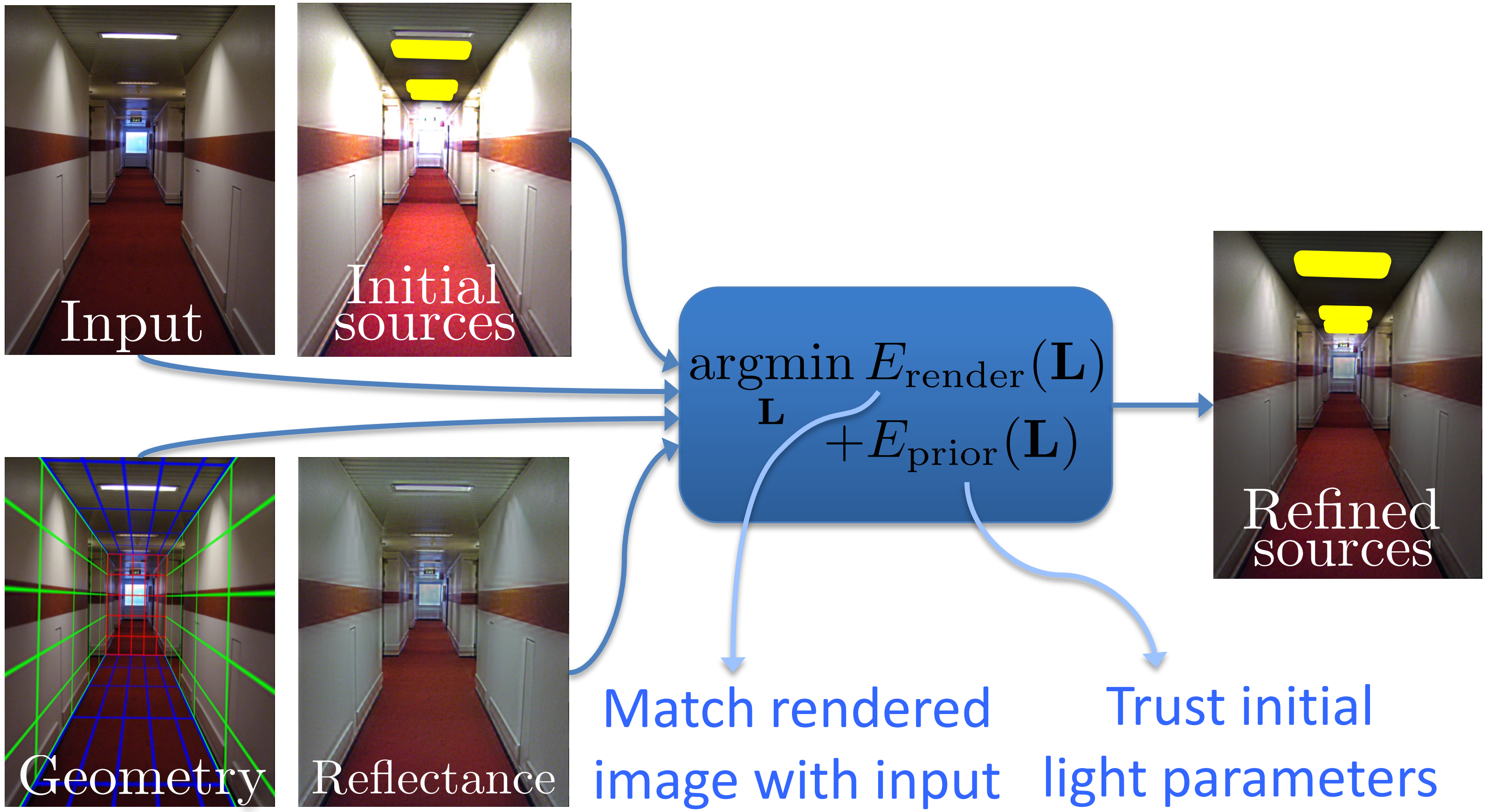


Lighting estimation



Light refinement

Match original image to rendered image

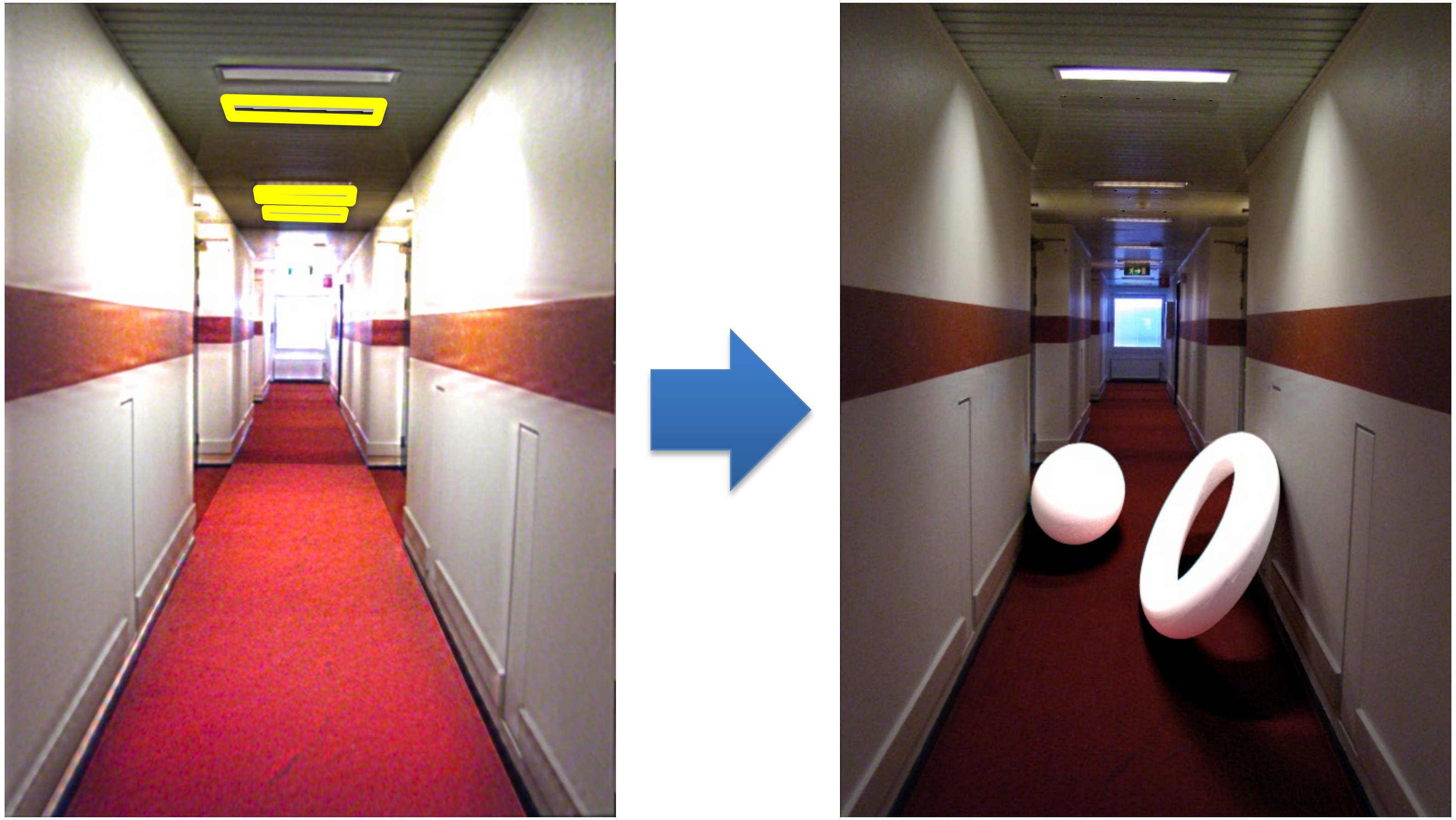


$$\underset{\mathbf{L}}{\operatorname{argmin}}\, E_{\text{render}}(\mathbf{L}) + E_{\text{prior}}(\mathbf{L})$$

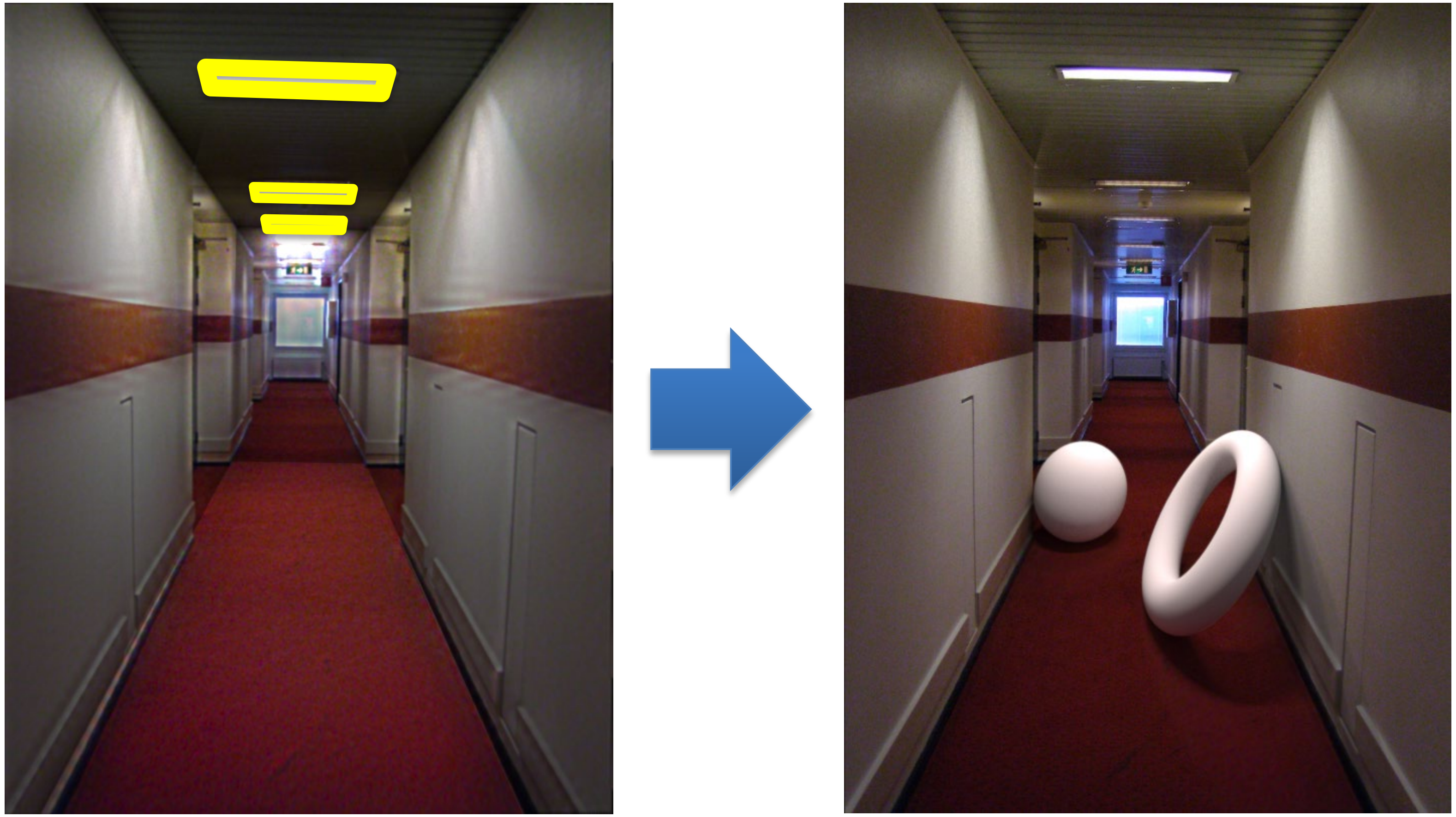
Match rendered image with input

Trust initial light parameters

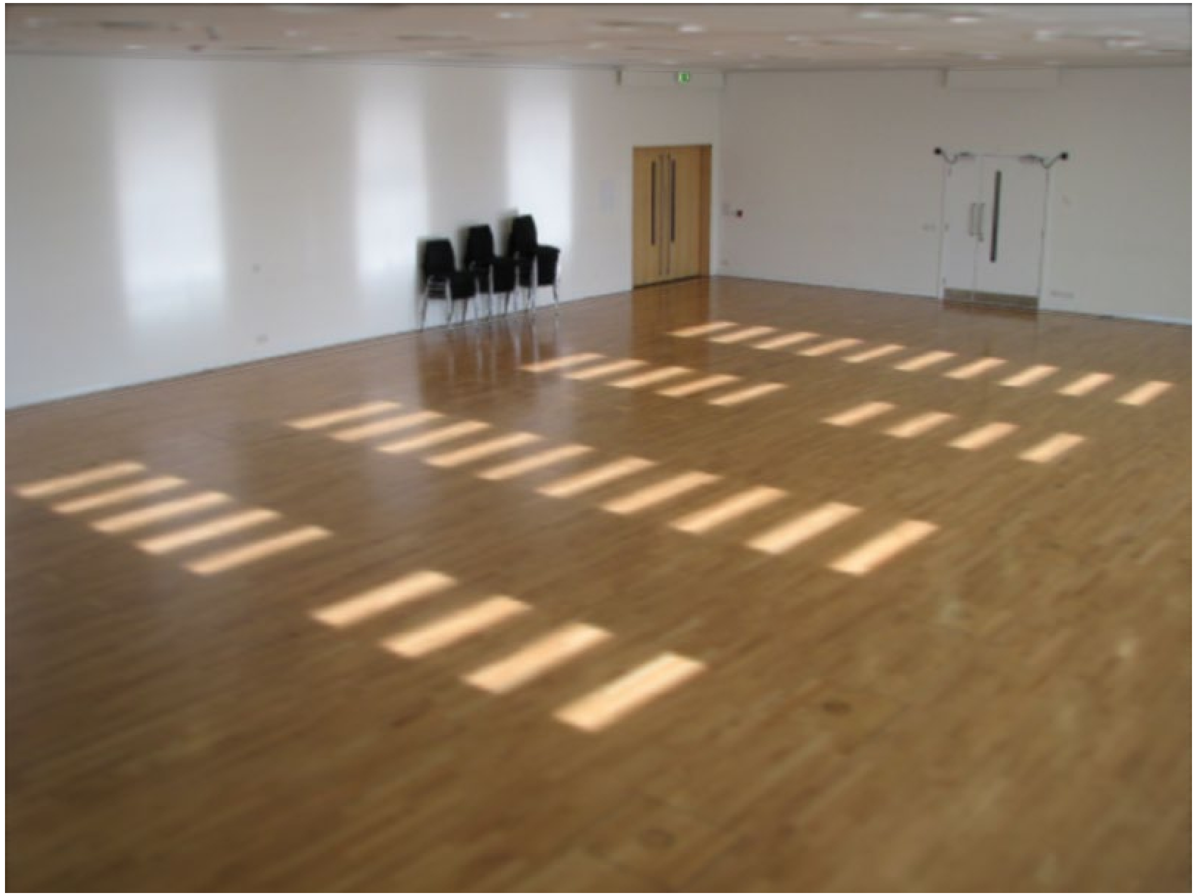
Initial light parameters



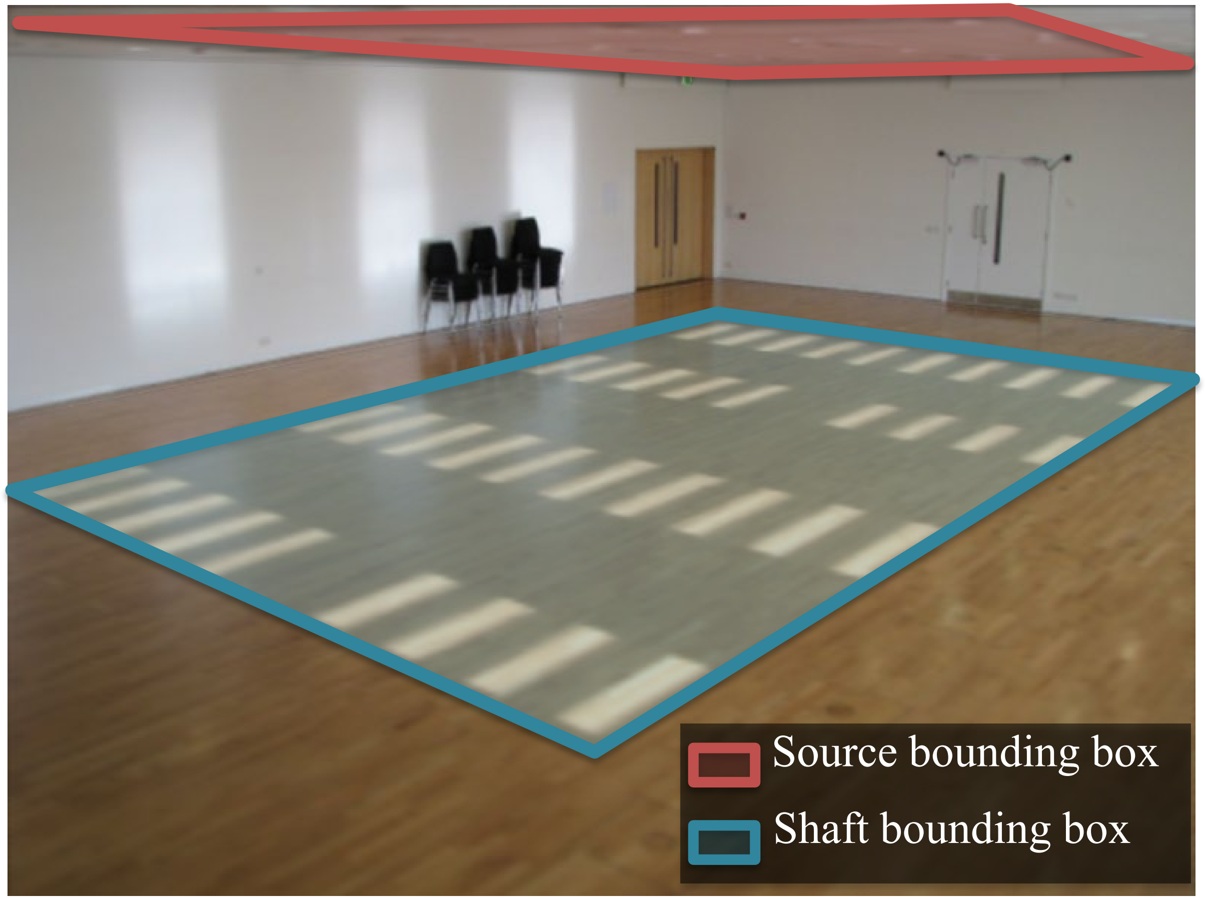
Refined light parameters



External light shafts

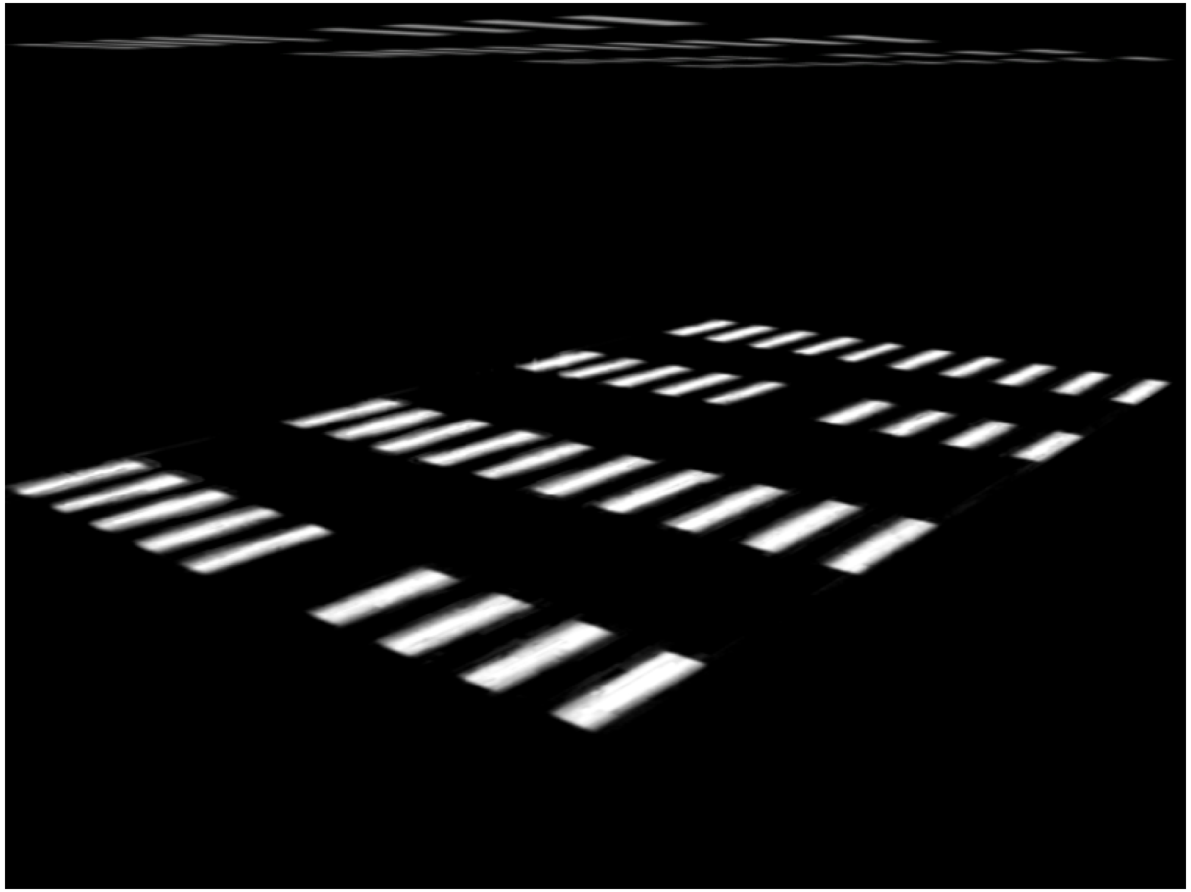


External light shafts

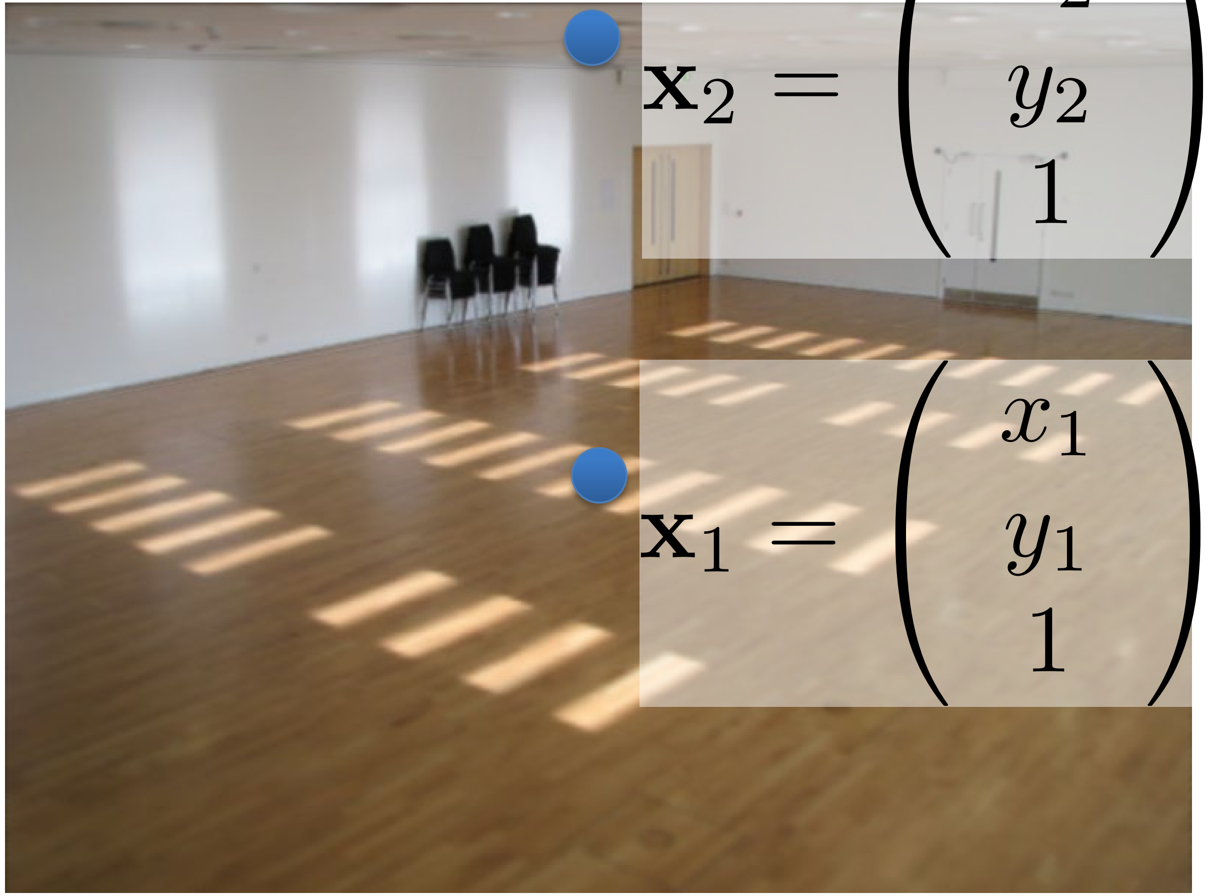


External light shafts

Shadow matting via Guo et al. [2011]



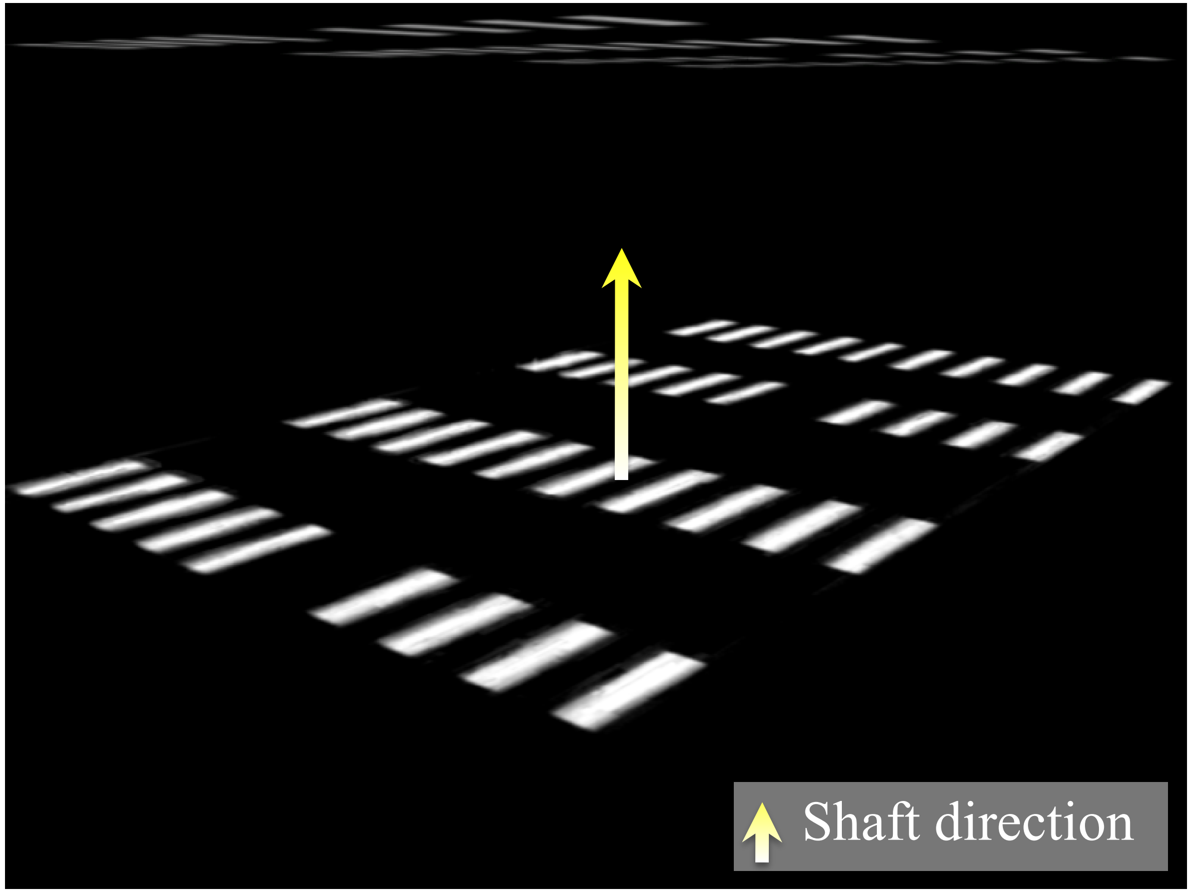
Setting light shaft direction



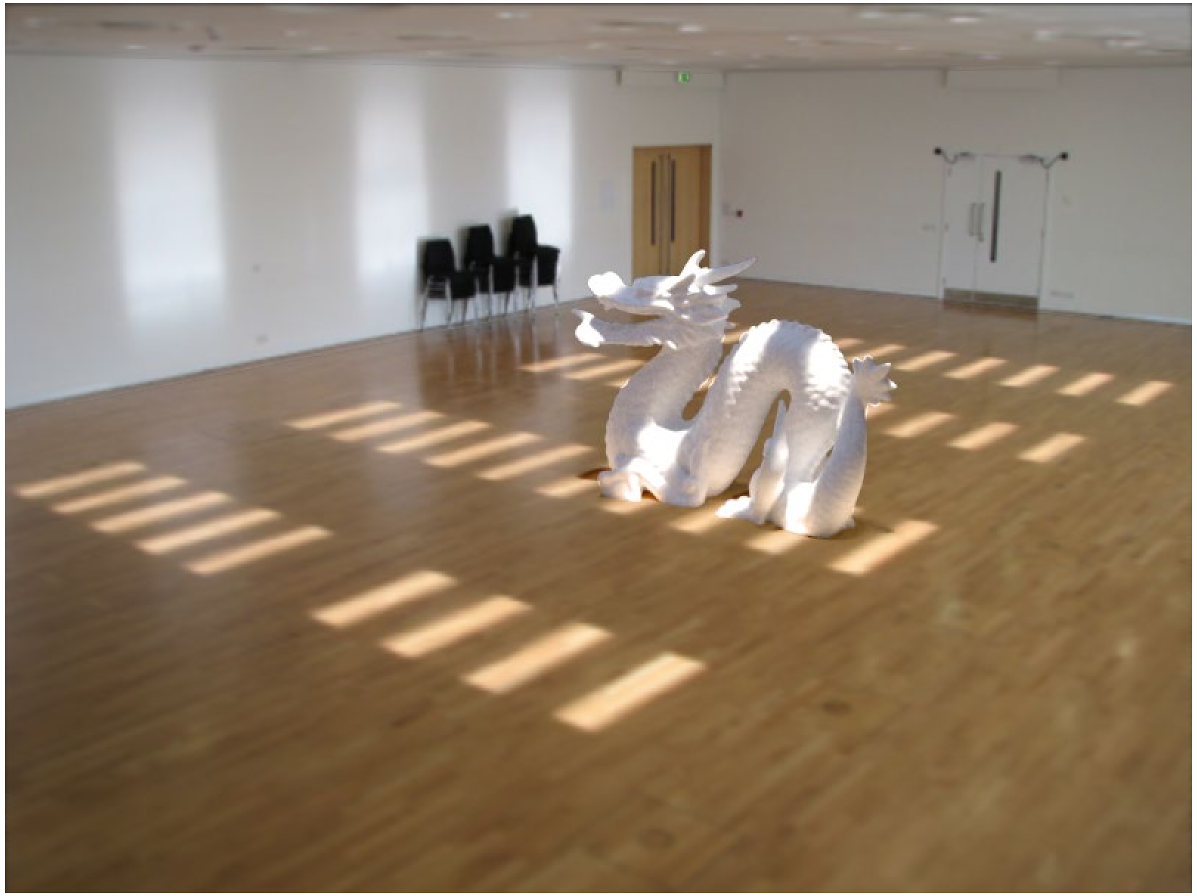
$$\mathbf{x}_2 = \begin{pmatrix} x_2 \\ y_2 \\ 1 \end{pmatrix}$$

$$\mathbf{x}_1 = \begin{pmatrix} x_1 \\ y_1 \\ 1 \end{pmatrix}$$

External light shafts



Light shaft result



Inserting objects

- Representation of geometry, materials and lights is now compatible with 3D modeling software
- Two methods of insertion/interaction
  - Novice: image space editing
  - Professional: 3D modeling tools (e.g. Maya)
- Scene rendered with physically based renderer (e.g. LuxRender, Blender's Cycles)

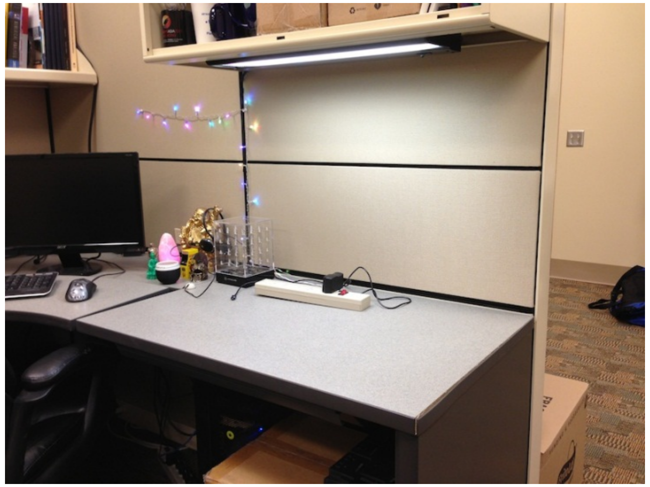
Blender demo

Final composite

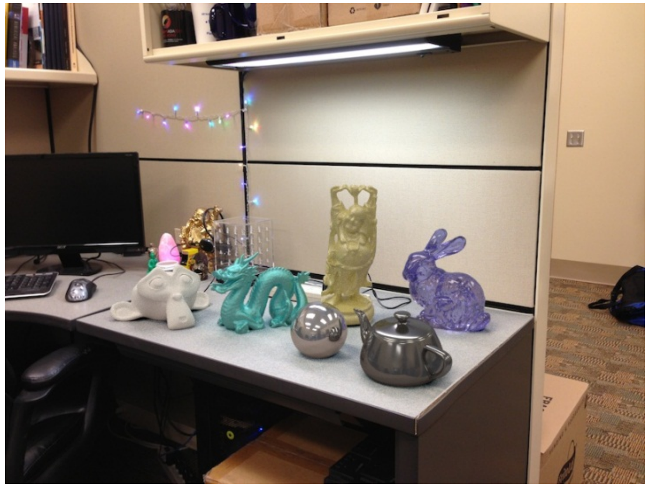
Additive differential technique [Debevec 1998]

composite = M.*R + (1-M).*I + (1-M).*(R-E)*c

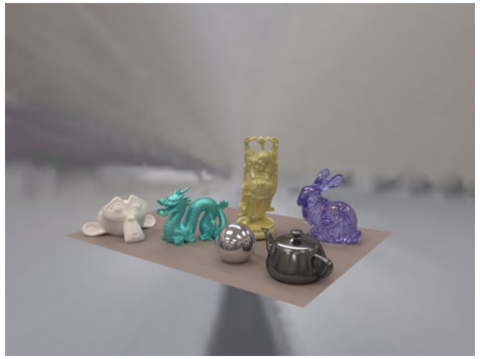
effect multiplier



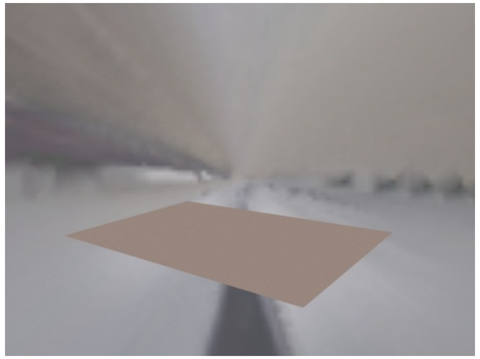
I (background)



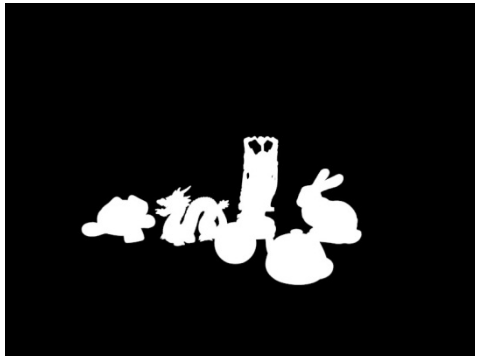
composite



R (rendered)



E (empty)



M (mask)

Putting it all together

Video

Research directions

- Can we do better with
  - Multiple images?
  - Videos?
  - Depth?
- Better scene understanding?
- How to insert image fragments?

Fully Automated Scene Modeling

Karsch et al. 2014: http://vimeo.com/101866891

Summary

- We can accurately predict how a 3D object would look in a depicted scene by recovering
  - Viewpoint: camera matrix, single view geometry
  - Scene geometry: single-view geometry
  - Material: “intrinsic image approaches”
  - Lighting: solve for lights such that rendering reproduces image
- Next classes: interest points, matching and alignment, and stitching