Claus B. Madsen
Architecture, Design and Media Technology
Aalborg University

DEVELOPING AND TESTING OUTDOOR DAYLIGHT ESTIMATION FOR AUGMENTED REALITY

Goal: realistic Augmented Reality

Overview of presentation

- Goal
- Overview of presentation
- Application context
- Fundamental problem and some constraints
- Overview of approach
- Shadow detection
- Illumination estimation
- Experimental results
- Conclusions and future work

Example application context

AR systems developed for “clients”

The problem is ill-posed

“APPEARANCE = REFLECTANCE x ILLUMINATION”

State-of-the-art…

- Pre-acquired Light Probe images (HDR environment maps captured with reflective sphere or fish-eye lenses)
- Placing objects with known geometry and reflectance (diffuse or glossy) properties in the scene
- Special cases (e.g., manually identifying vertical structures and their shadows in the image)

Our goal

Estimate illumination

- In general scenes
- In real-time
- With no special purpose objects
- Directly from image measurements
- With no HDR requirements

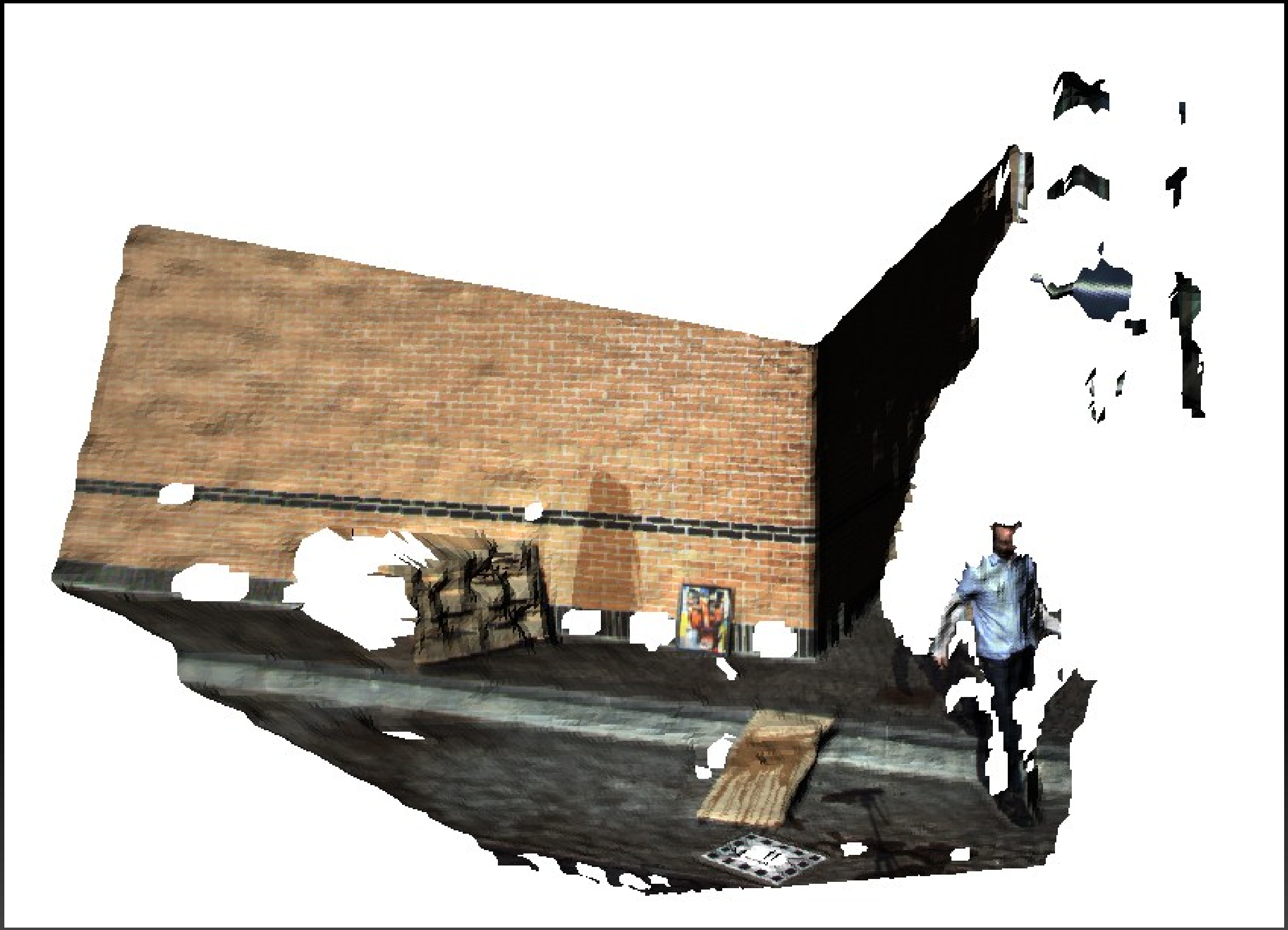
Applied constraints/assumptions

- Live 3D scene data from commercial stereo
- Outdoor daylight conditions
- Scene is geo-located
- Surfaces in scene are predominantly diffuse
- There are dynamic objects in the scene

Illumination?

- Radiance of sky (RGB) and radiance of sun (RGB)

Real-time 3D scene data



PointGrey Bumblebee XB3 stereo camera

Where does RADIANCE come in?

We use RADIANCE for all development, validation and testing…

Disclaimer: There is no spiffy RADIANCE witch-craft in this presentation, just an alternative application

Overview of approach

Step 1: Combine color and depth information to detect dynamic shadows (DYN3DIM 2009)

T- ΔT

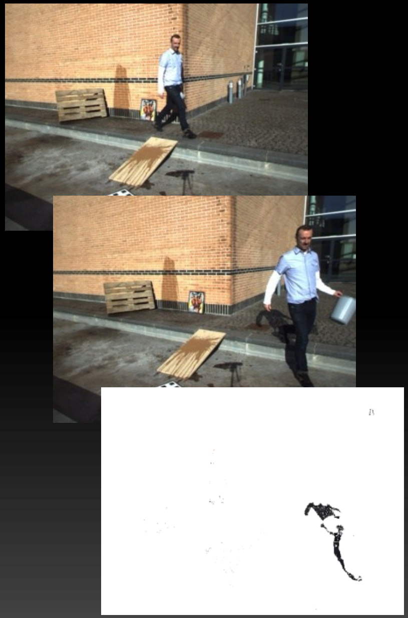
T

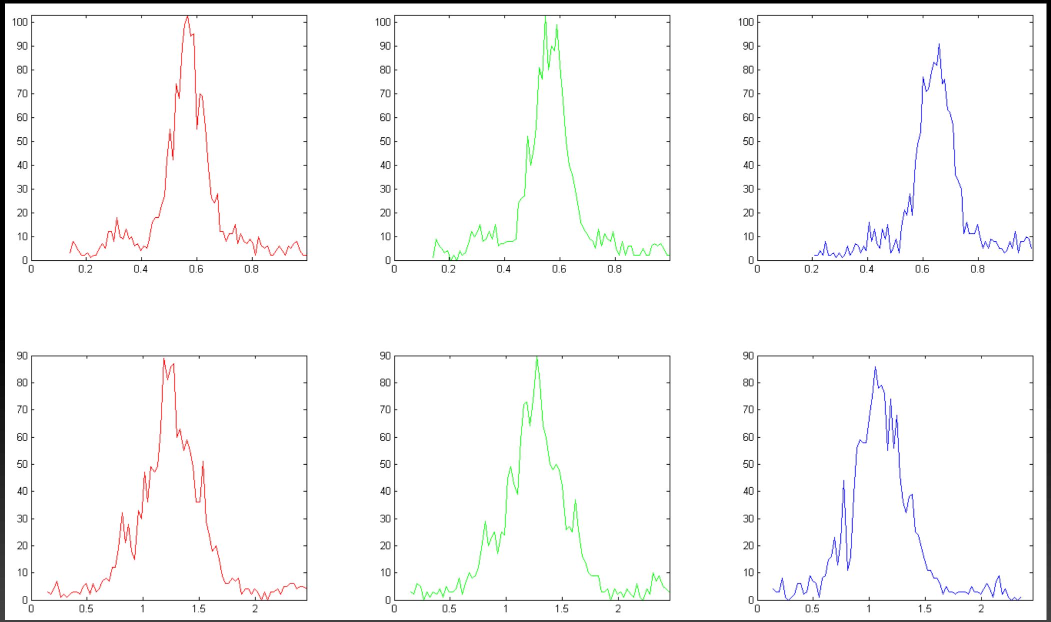
Shadow at time T

Objective: find pixels that are in shadow now, but were in sunlight a few hundred milliseconds ago…

Overview of approach (cont.)

Step 2: Use shadow and non-shadow versions of same pixel to vote for sky and sun radiance (GRAPP 2011)





Sun and sky radiance histograms

Overview of approach (cont.)

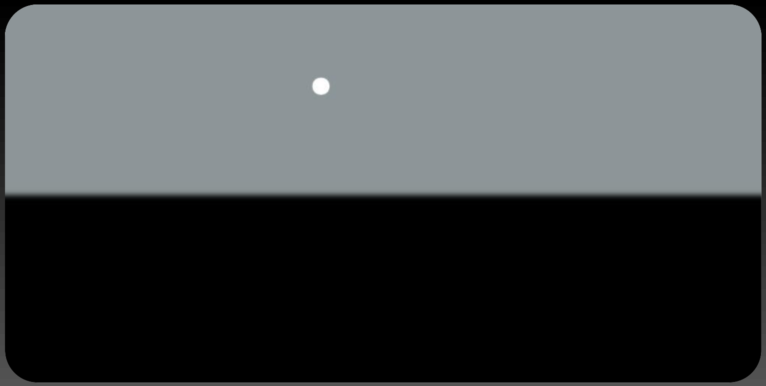
Step 3: Composite a “world” from:

a) textured real scene mesh
b) augmented object mesh
c) sky dome with estimated radiance
d) sun disk with estimated radiance

... And render using differential rendering (Debevec, SIGGRAPH 1999)

Example sequence

Example



Illumination estimation in more detail

A pixel in sunlight (diffuse surface and linear camera):

$$P(T-\Delta T) = c \cdot \frac{1}{\pi} \cdot \rho \cdot \left(V_a(T-\Delta T) \cdot E_a(T-\Delta T) + \vec{n} \cdot \vec{s} \cdot E_s(T-\Delta T)\right)$$

A pixel in shadow:

$$P(T) = c \cdot \frac{1}{\pi} \cdot \rho \cdot V_a(T) \cdot E_a(T)$$

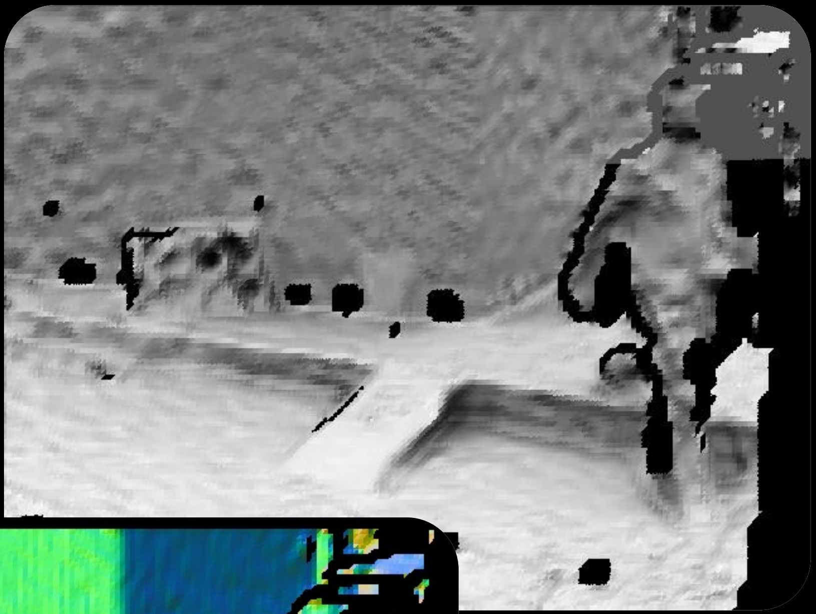
The special sauce: Shadow-to-sun *ratios*

$$C = \frac{P(T)}{P(T - \Delta T)}$$

Using such ratios eliminates the scaling factor from the camera, *and* the surface reflectance…

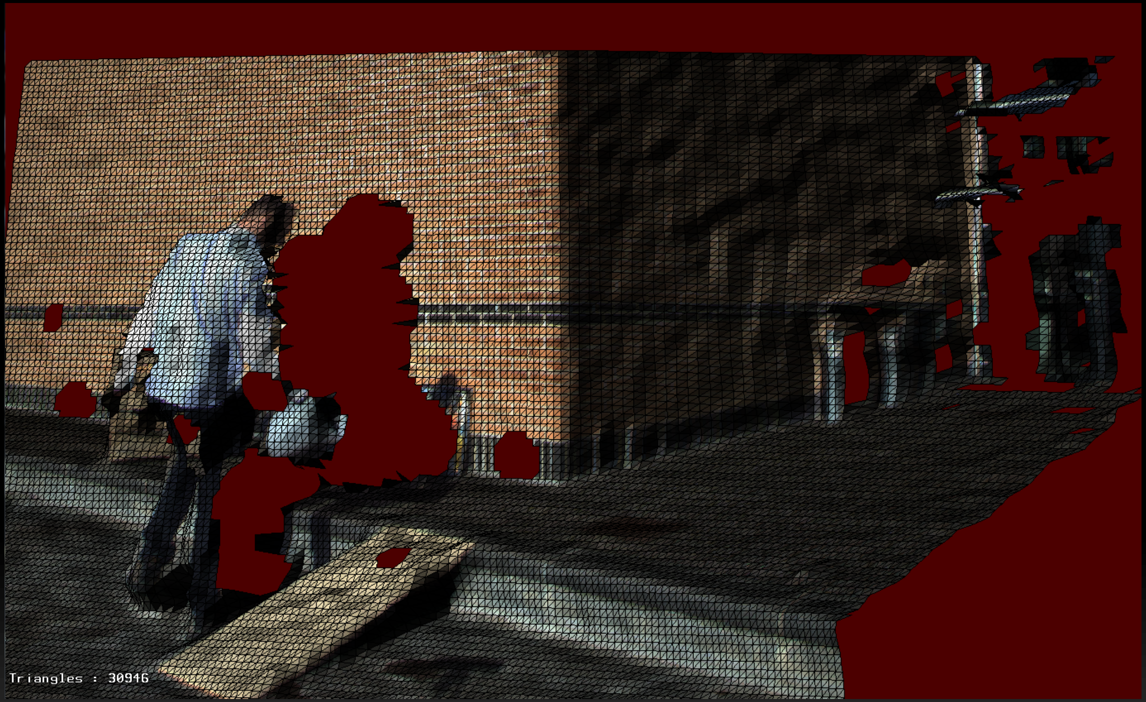
Per pixel geometry factors

Ambient occlusion map



Normal map

Back-projecting the image onto the geometry in RADIANCE



```
{
  PICTPROJECT.cal for image: E:\sequences\seq3\calibration\139color.ppm
  Autogenerated by realpose.m, a MATLAB camera calibration program
  by Claus B. Madsen, CVMT/AAU, copyright 2011
}

{The row vectors in the world to camera rotation matrix}
camxvec(i) : select(i, -0.63395514, -0.77326564, -0.01269386);
camyvec(i) : select(i, 0.11371990, -0.10944244, 0.98746652);
camzvec(i) : select(i, -0.76496318, 0.62456593, 0.15731729);
translation(i) : select(i, -0.41594783, -1.03067795, -2.40969019);

{Transform ray intersection point to camera coordinates}
Pxcam = dot(P,camxvec) + translation(1);
Pycam = dot(P,camyvec) + translation(2);
Pzcam = dot(P,camzvec) + translation(3);

{Focal length yielding image plane from -0.5 to +0.5 in smallest FOV}
minfovnormalizedfocallength = -0.90222759;

{Compute perspective projection of point onto normalized image}
{and translation to image coord system origin in lower left corner}
ratio = 1.33333333;
u = minfovnormalizedfocallength * Pxcam / Pzcam + 0.5*ratio;
v = minfovnormalizedfocallength * Pycam / Pzcam + 0.5;

{Create Boolean which is 1 inside image, 0 outide}
inpic = and(and(if(ratio - u, 1, 0), if(1 - v, 1, 0)), and(if(u, 1, 0), if(v, 1, 0)));
```

Direct computation of sky and sun irradiances

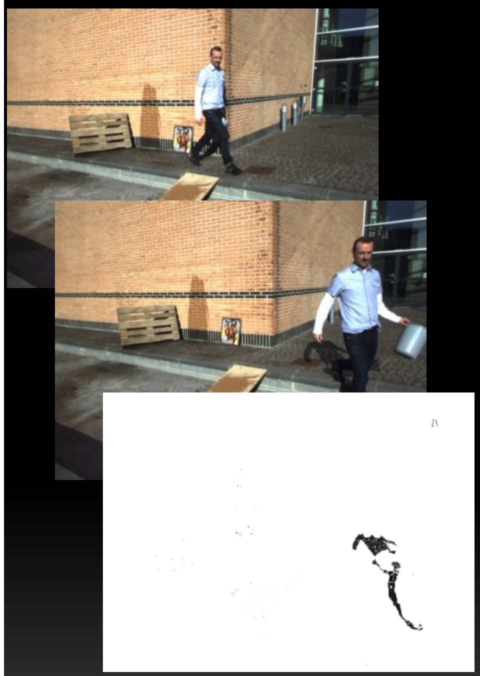
Sky irradiance contribution

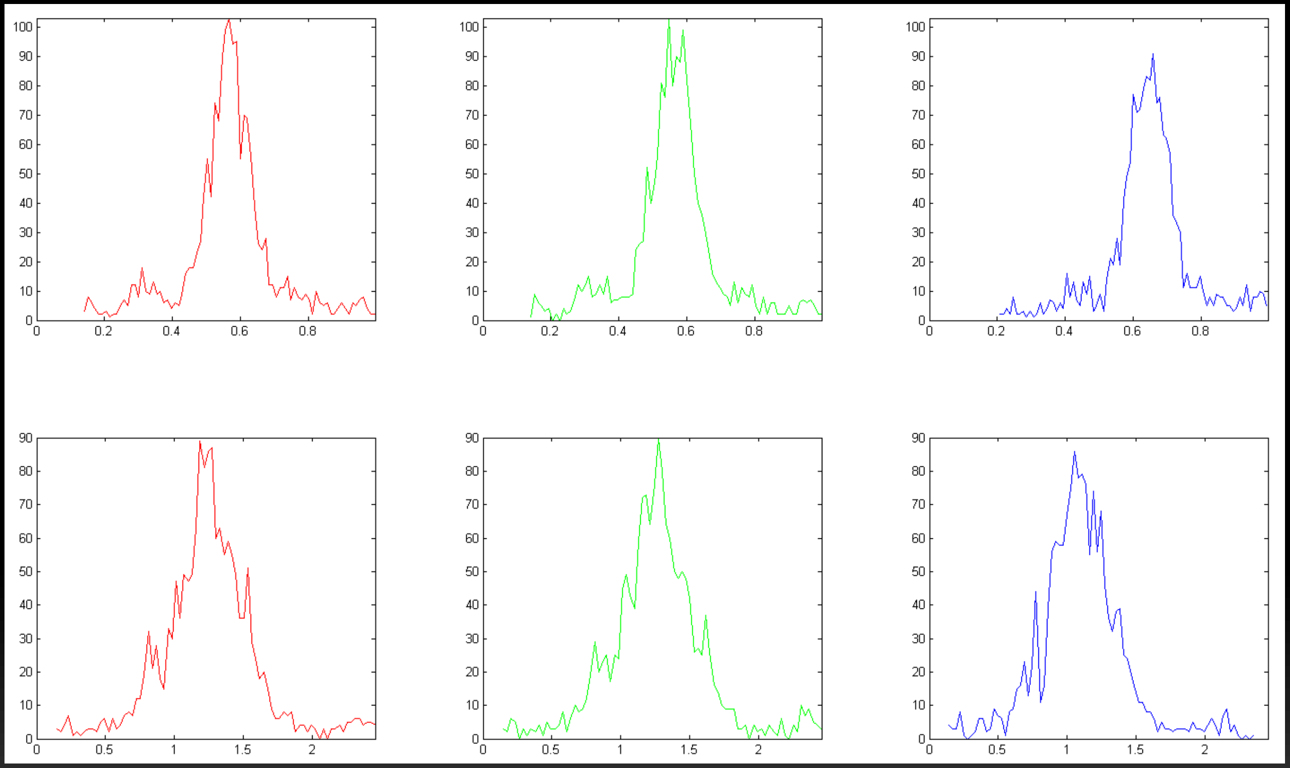
$$E_a = f_a(V_a, \vec{n}, \vec{s}, C)$$

Sun irradiance contribution

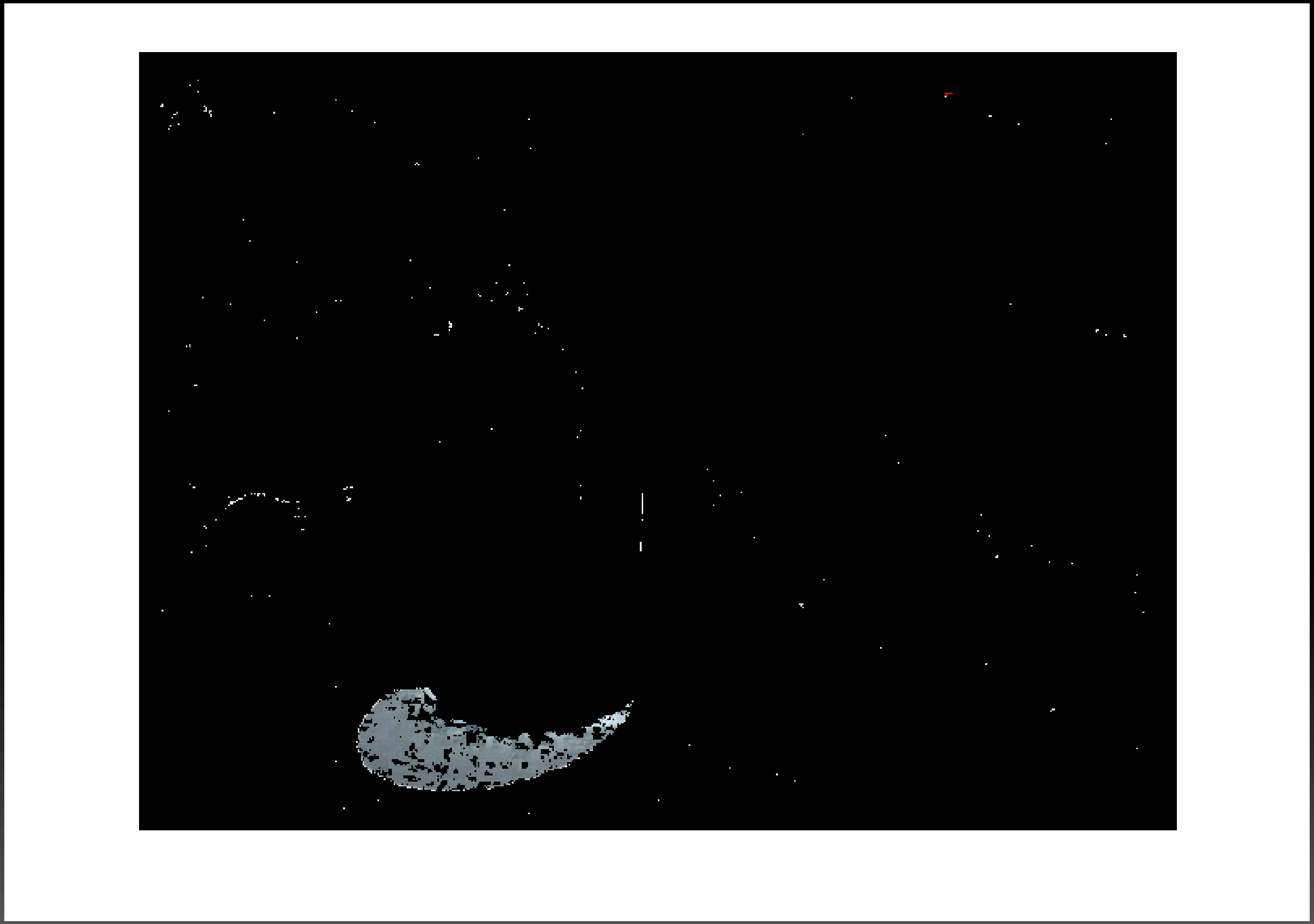
$$E_s = f_s(E_a, V_a, \vec{n}, \vec{s}, C)$$

Shadow pixels vote for illumination



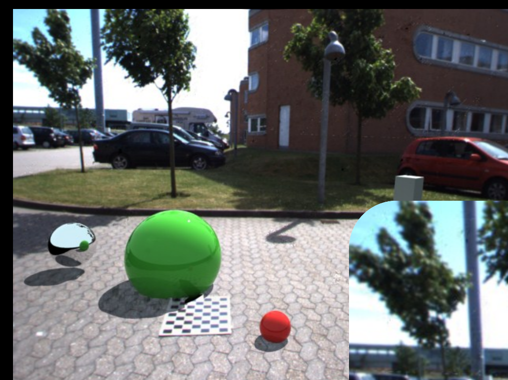
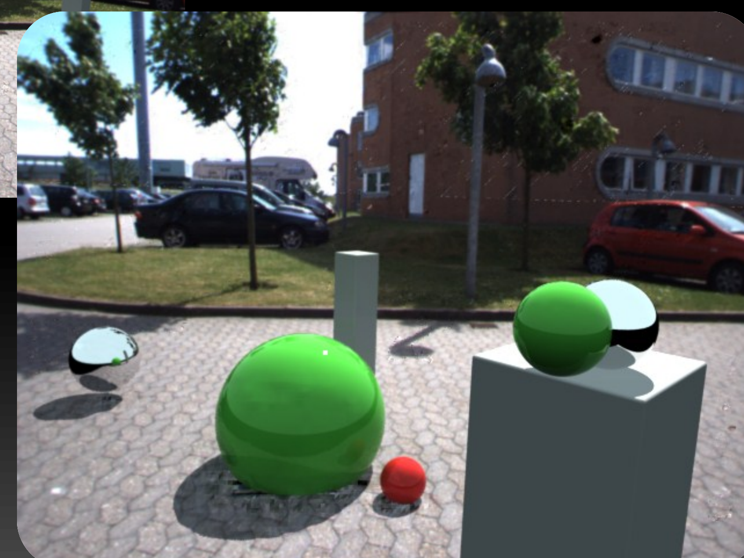


Shadow pixels vote for illumination



Illumination estimation:
real geometry and reflectance , synthetic illumination



Estimated sun and sky radiances are within 5% of ground truth

Main source of error: indirect (global) illumination

Another example…

Conclusions

- No special purpose objects
- No HDR requirements
- Follows camera AGC
- Non-iterative
- Potential for real-time

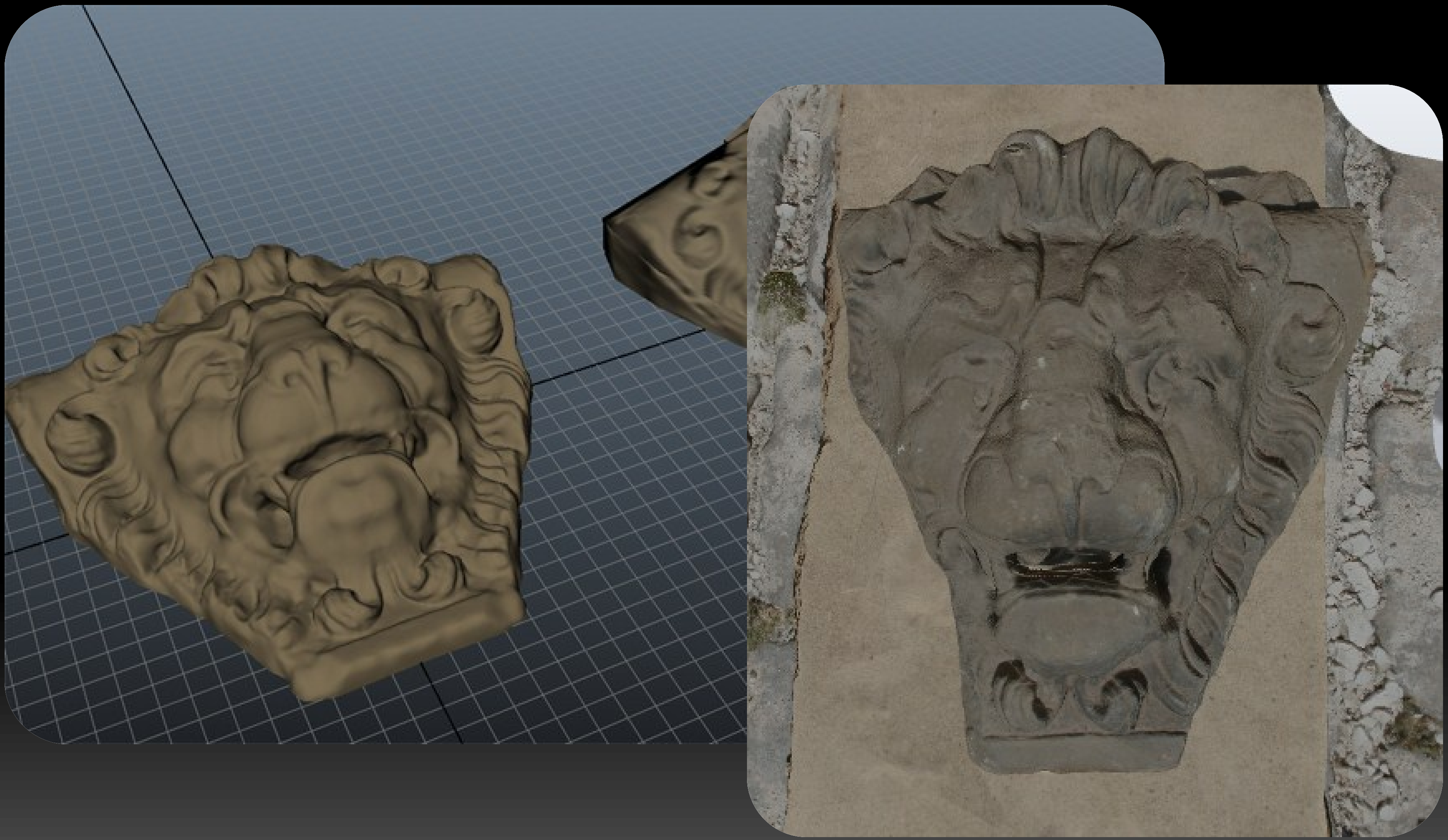
Future (current) work

1. Extend to handle Global Illumination contribution
2. Extend to function in overcast conditions and without dynamic shadows

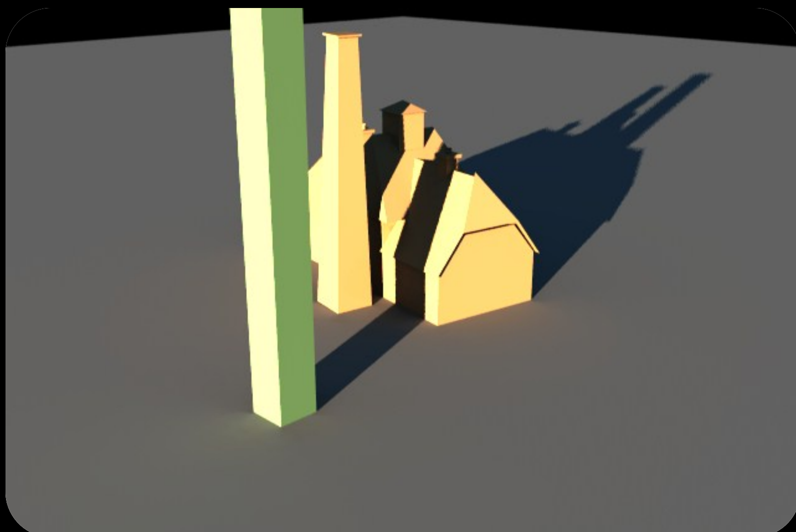
Global illumination contribution

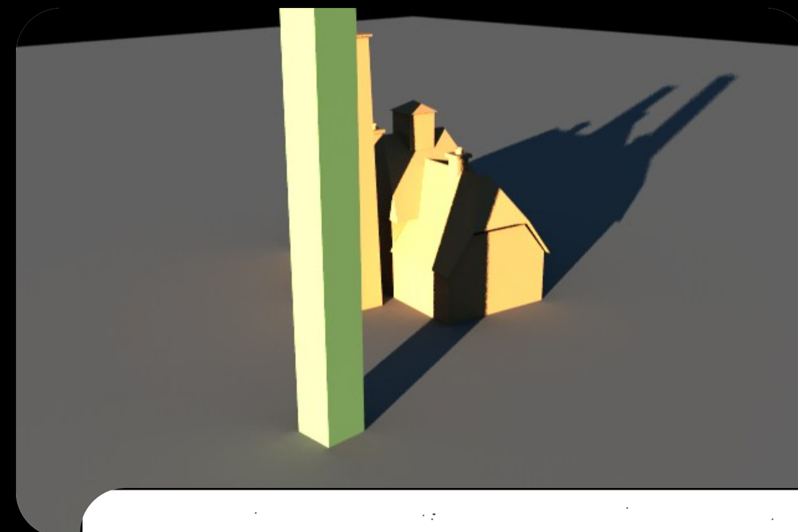
Thank you for your attention

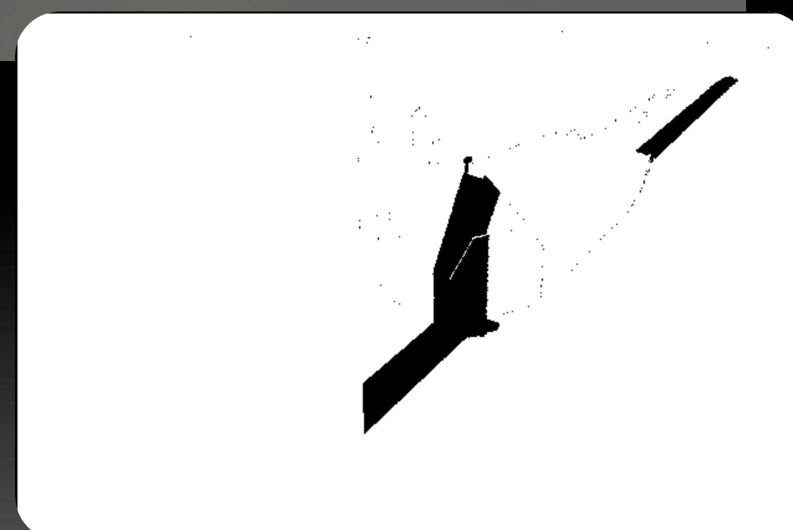
123D Catch



Illumination estimation:
synthetic geometry, reflectance and illumination







Estimated sun and sky radiances are within 5% of ground truth

Main source of error: indirect (global) illumination

It doesn´t have to be a person ☺