

Relighting People

for virtual production: light, shape, material,
and a live pipeline with Sapiens2, 3DGS and
Unreal Engine

Yoshihiro Kanamori

University of Tsukuba, Japan

HSD Summer School 2026 · Hochschule Düsseldorf



It started with a wish to travel



...but travelling costs a lot of money



Me



My colleague, Yuki Endo

Our idea: travel virtually!

AHA!



Virtual travel means relighting

- One photo of a person
- Many backgrounds
- The light on her **changes to match**



Today's journey

1 Why do we need relighting?

2 What does relighting need?

3 Light, shape and material

4 How film productions do it

5 Relighting with simple capture

6 Neural relighting research

7 Our approach this week



1

Why do we need relighting?

Compositing, film, LED volumes and photography

Just pasting a person looks wrong



Person

+



Background

=



Composite

Unnatural!

Matching the light makes it natural



Relit person

+



Background

=



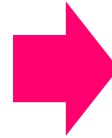
Composite

Natural!

Hollywood shoots actors on green screens (chroma keying)

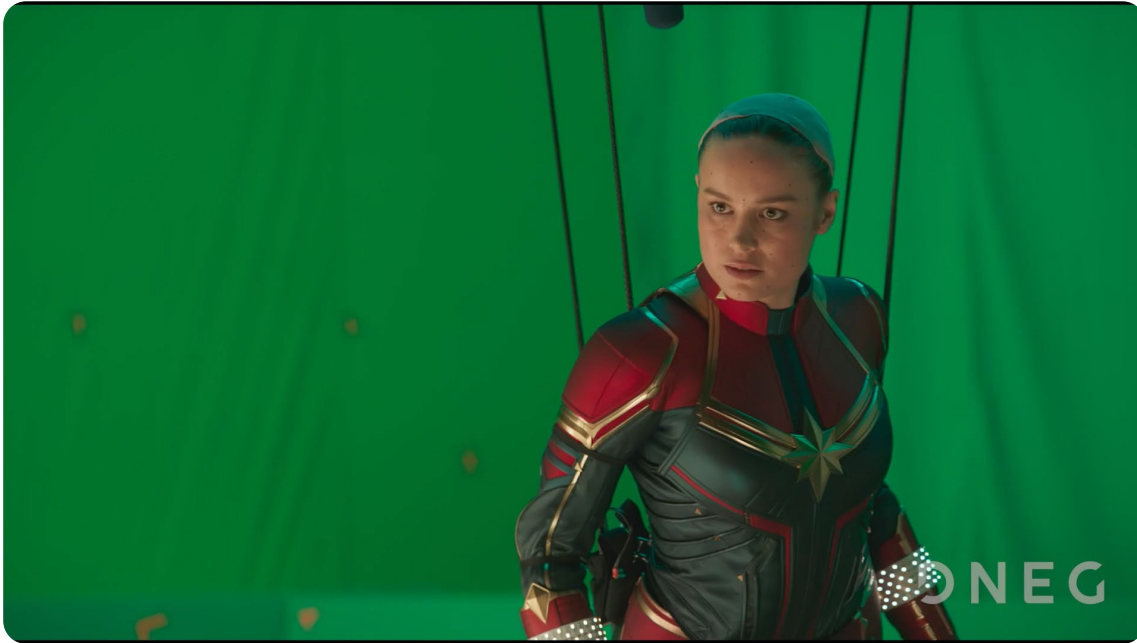


On set

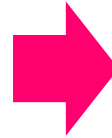


Final shot

...and adjusts the light afterward



On set



Shading and material adjusted

LED volumes: the wall is the lamp

- A curved LED wall shows an Unreal Engine world
- The wall **emits light** onto the actor
- Correct light and reflections in camera
- But: very expensive stages



▶ Click the image to watch

Dark Bay: an LED studio in Germany

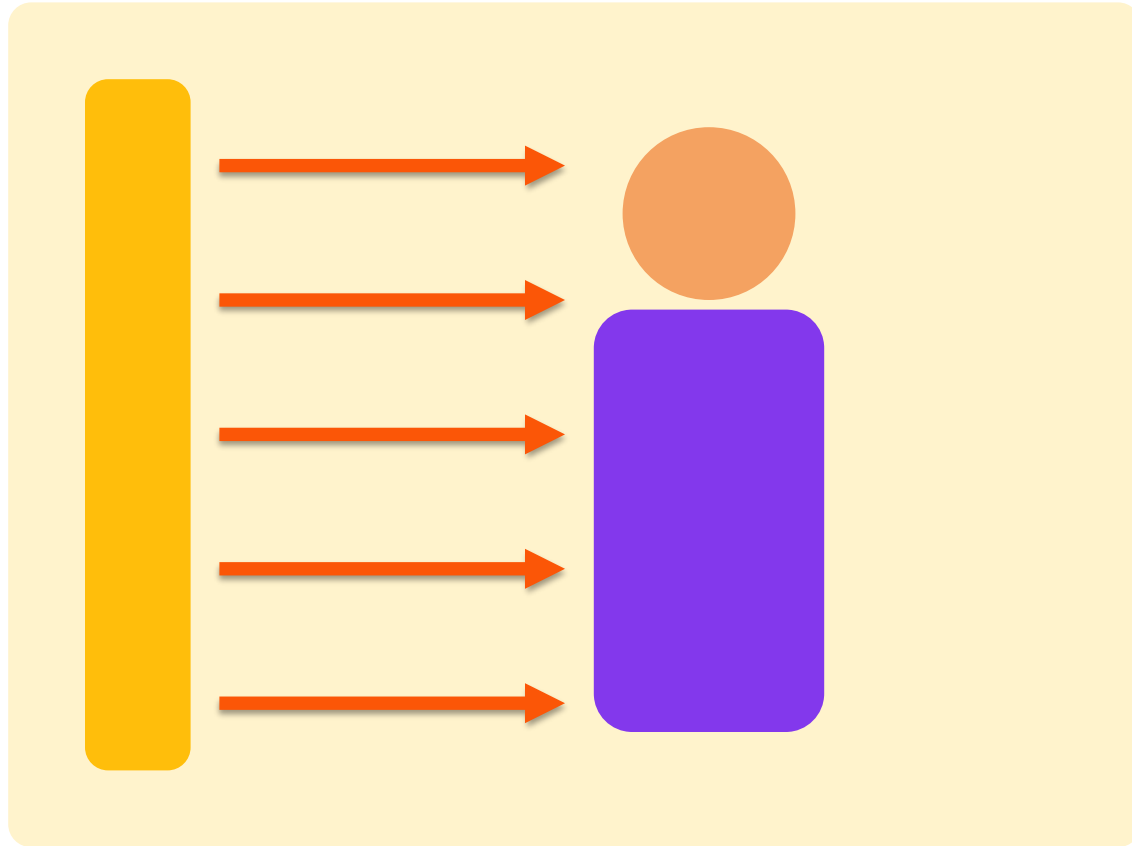
- Studio Babelsberg, near Berlin
- Built for Netflix's '1899'
- Unreal Engine worlds on ROE LED panels
- Permanent and costly



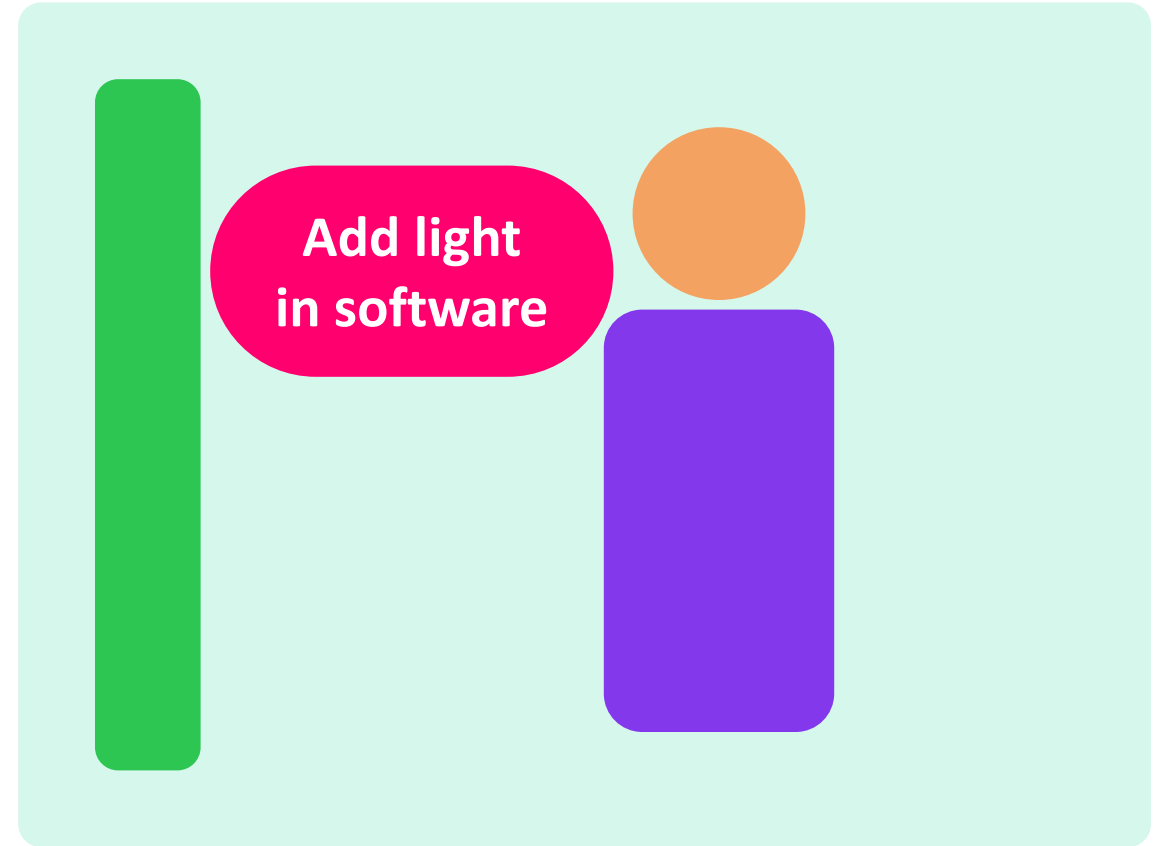
▶ Click the image to watch

In Depth Cine: The virtual production of 1899 (YouTube)

A green screen gives no useful light



LED wall: light for free



Green screen: no useful light

Photographers want relighting too



Studio photography: many lamps

Light moved after the shot

Right: relit with a commercial portrait editor (PortraitPro)

Relighting: decide the light afterward

Relighting = change the light of an image **after** it was captured



Film VFX



Green screen



Portraits



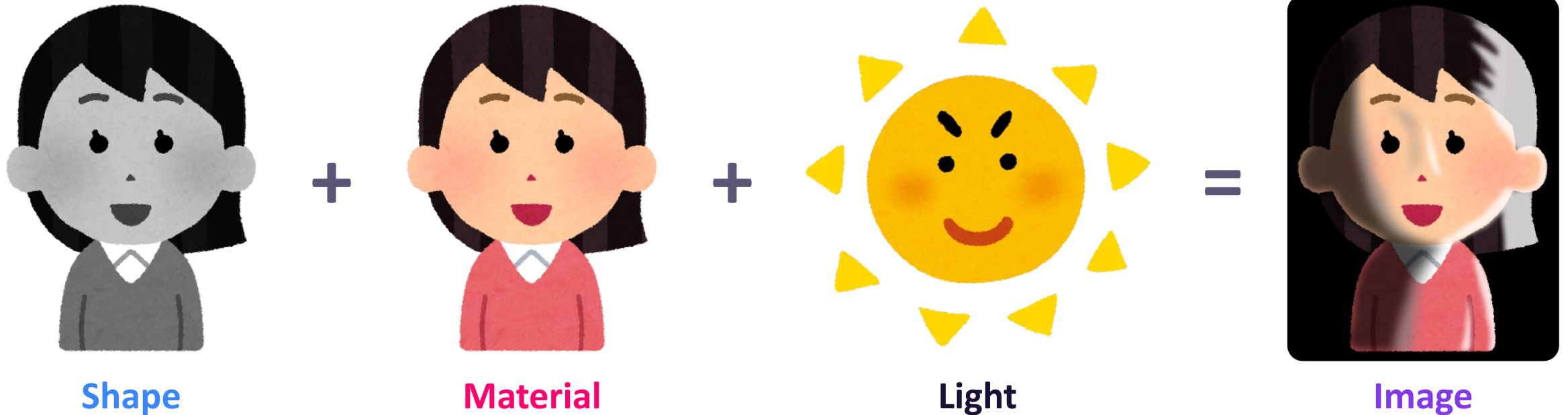
Virtual travel

2

What does relighting need?

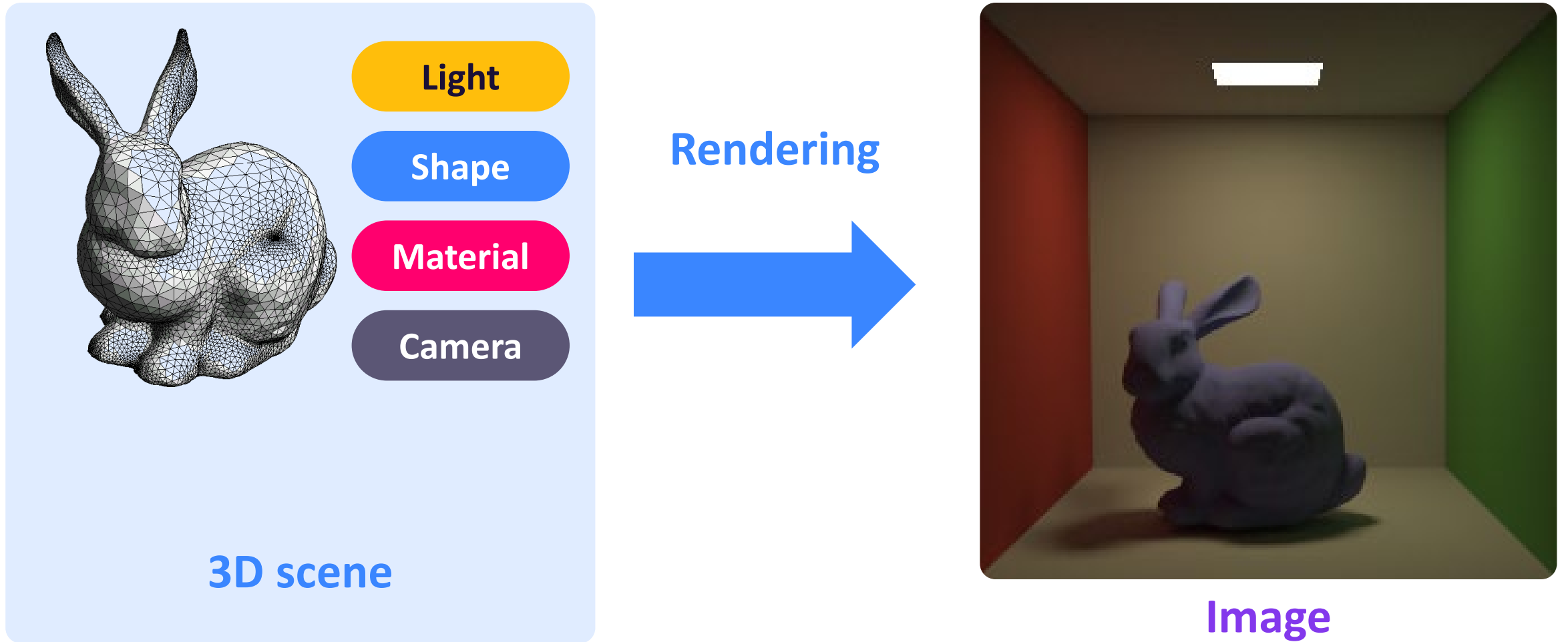
Shape, material, light, and why one photo is not enough

Three things decide how an object looks



Change only the **light**, keep shape and material: that is relighting

Rendering: from a 3D scene to an image



Inverse rendering: from an image back



Photo

Inverse rendering



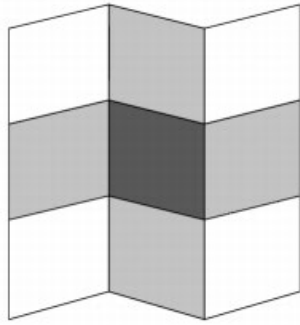
Light ?

Shape ?

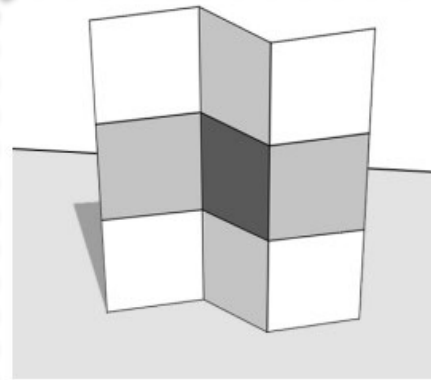
Material ?

What we need

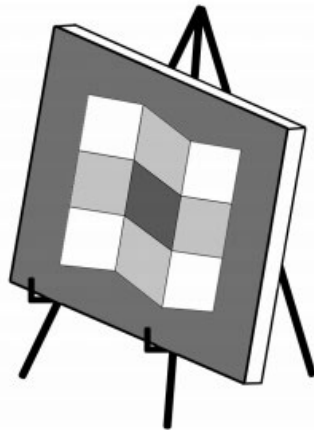
One image, many explanations



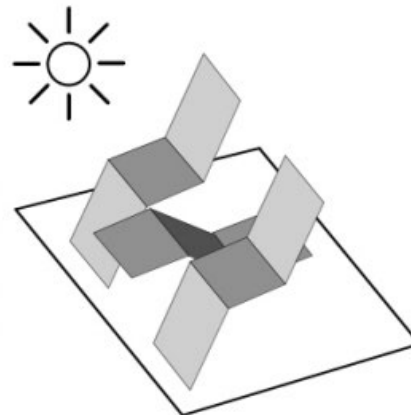
(a) an image



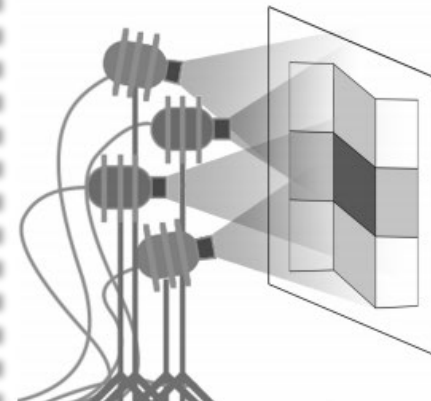
(b) a likely explanation



(c) painter's explanation



(d) sculptor's explanation



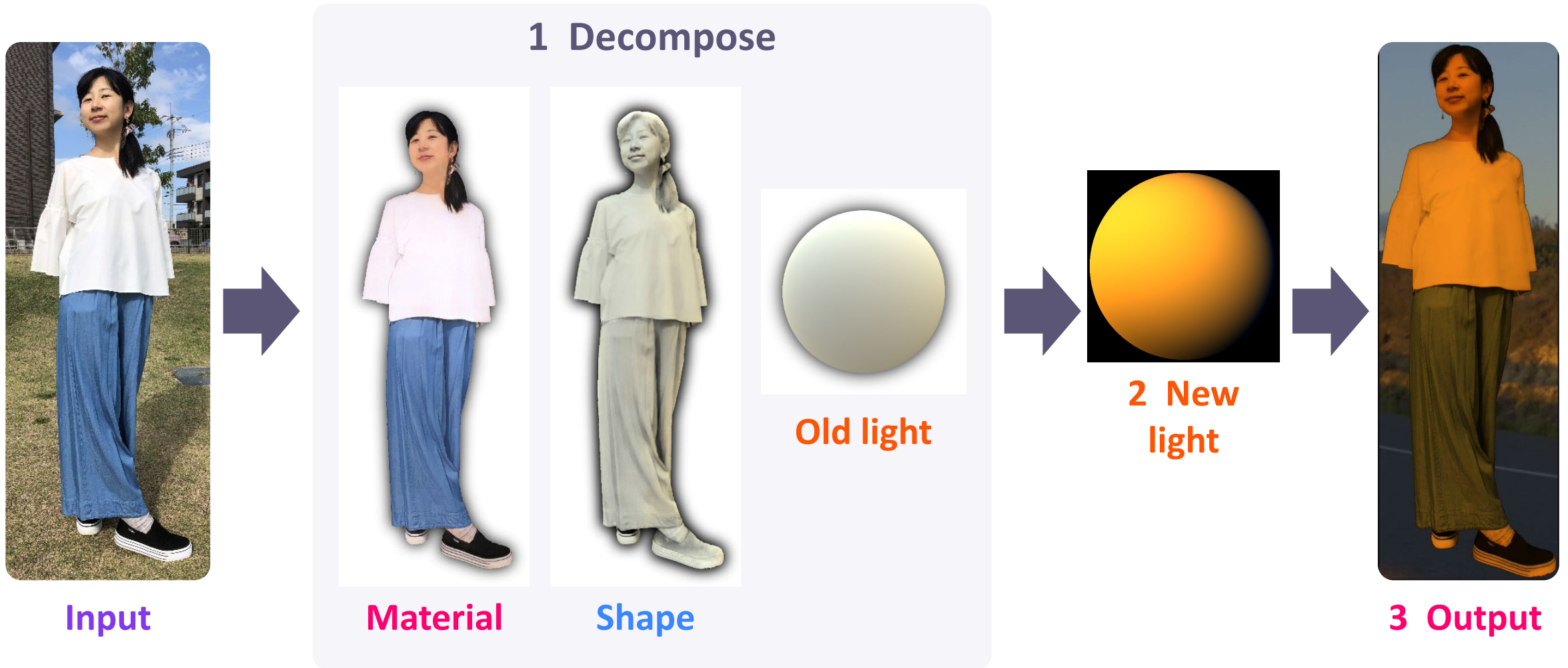
(e) gaffer's explanation

Color or light? The camera can't tell



Same pixels, two explanations → prior knowledge from data decides

Relighting = take apart, then rebuild

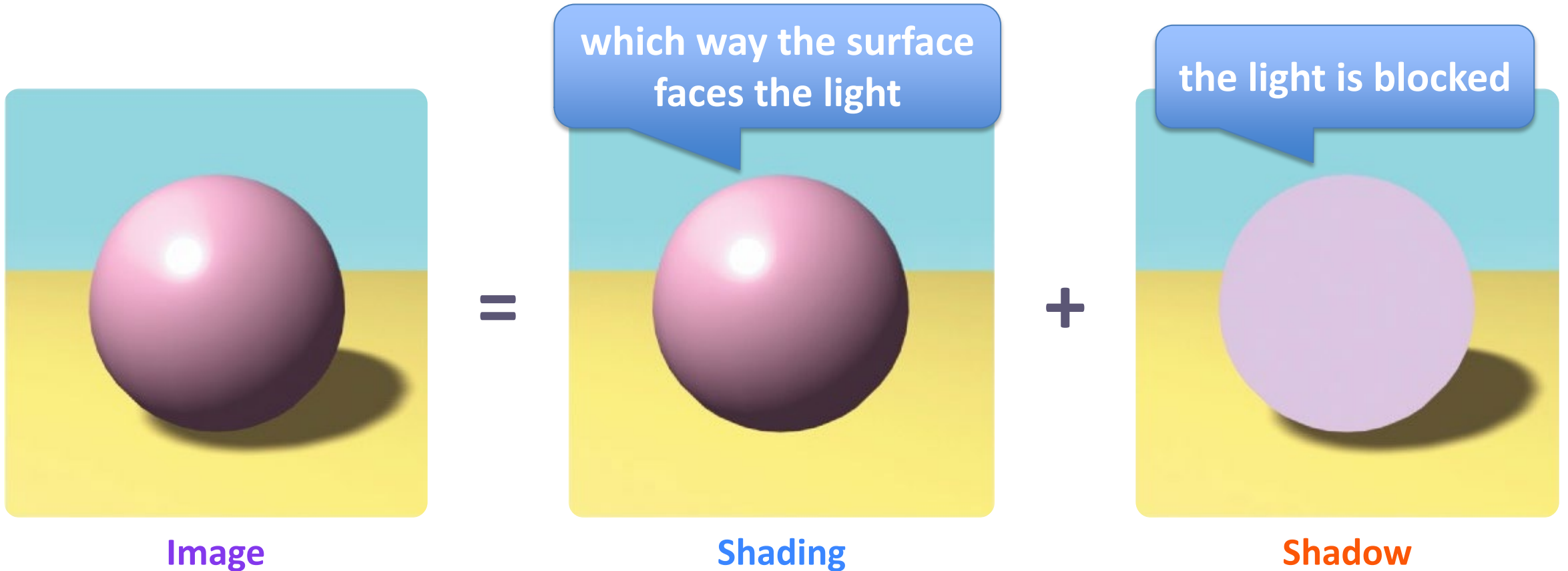


3

Light, shape, and material

The three ingredients, one at a time

Shading and shadows are different

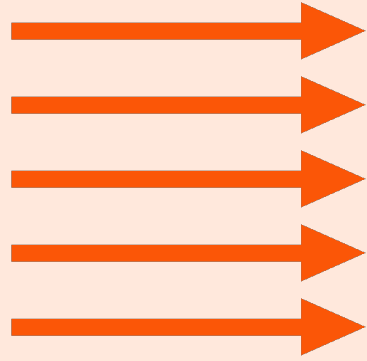


Four kinds of light in computer graphics



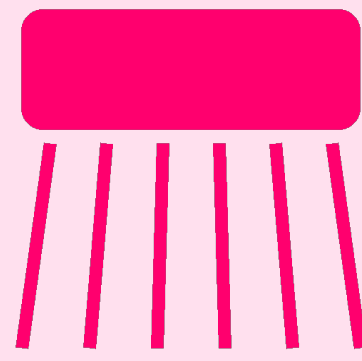
Point

A bare light bulb



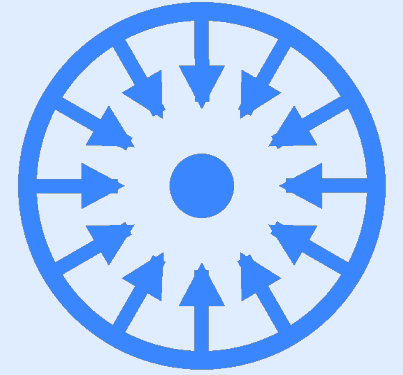
Directional

The sun



Area

A softbox, a window



Environment

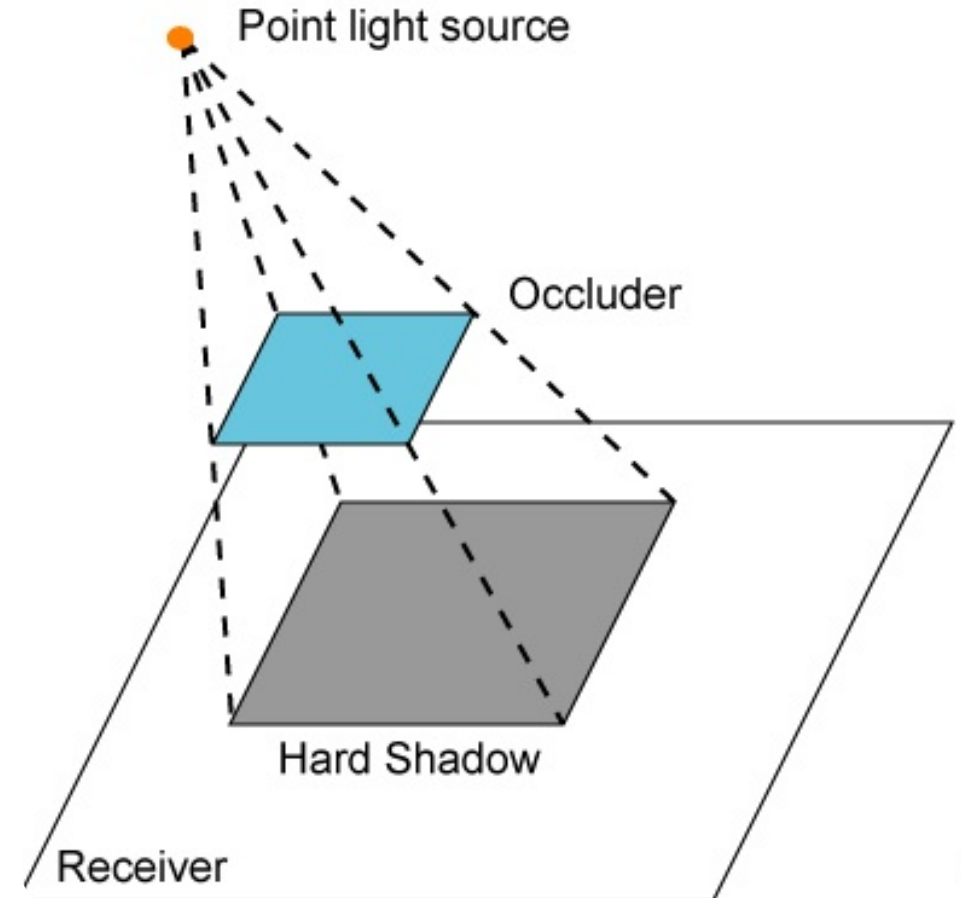
The whole sky

This week: area lights

Small lights cast hard shadows

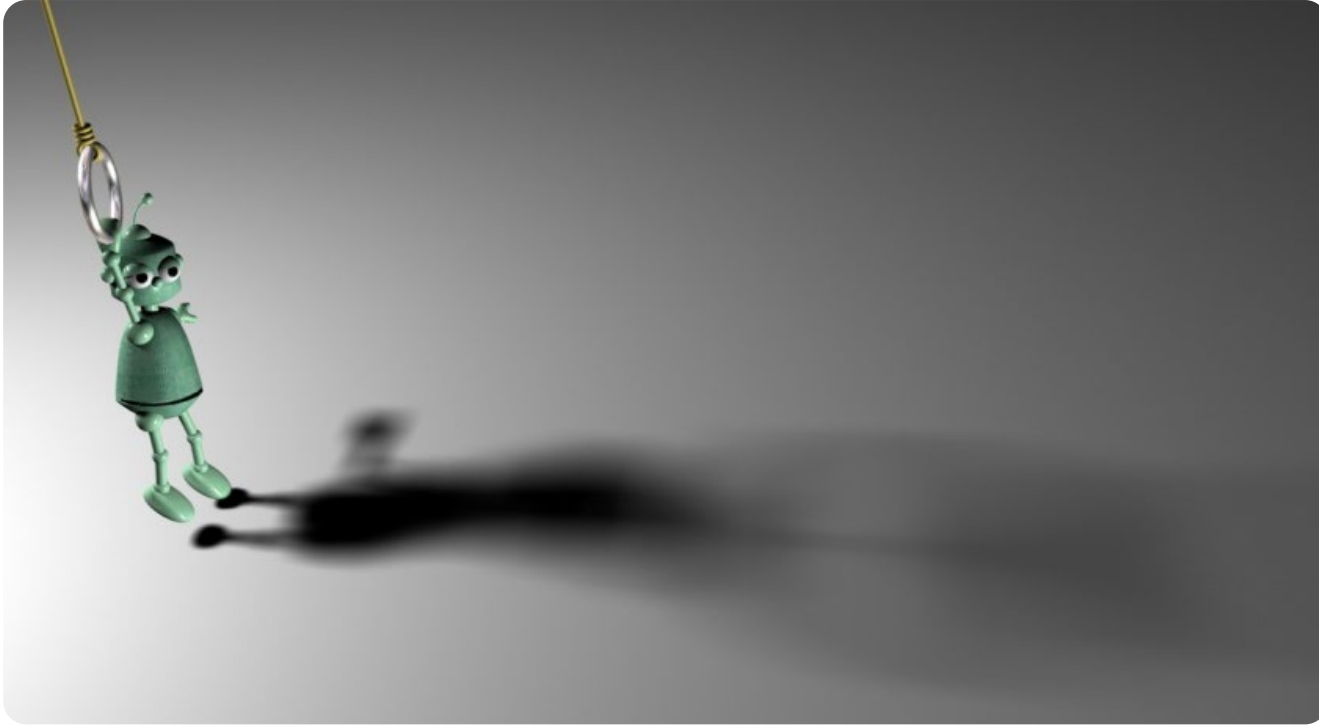


Sharp edge

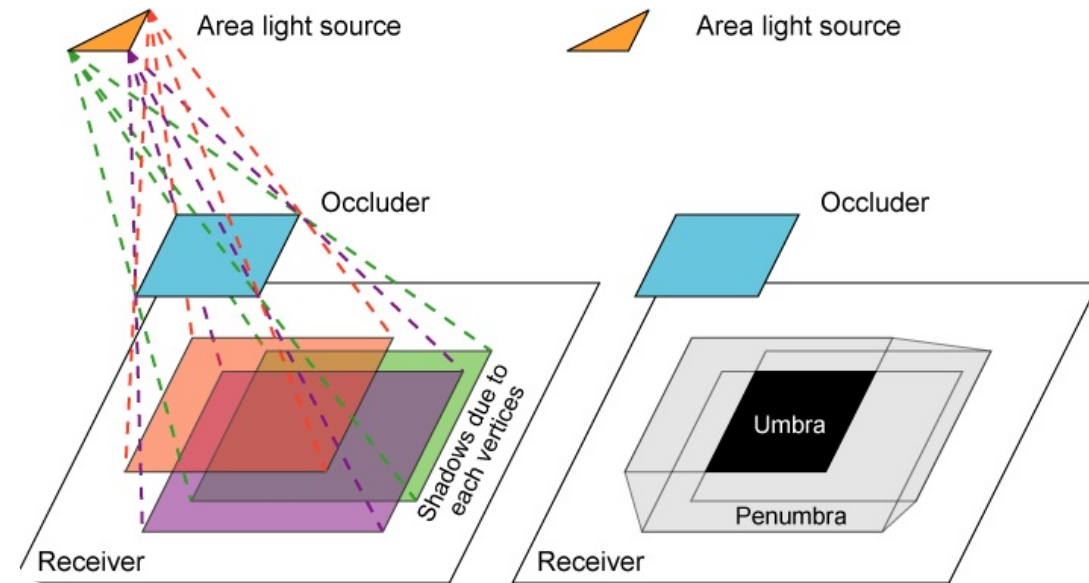


Point light: seen or not seen

Large lights cast soft shadows



Blurry edge

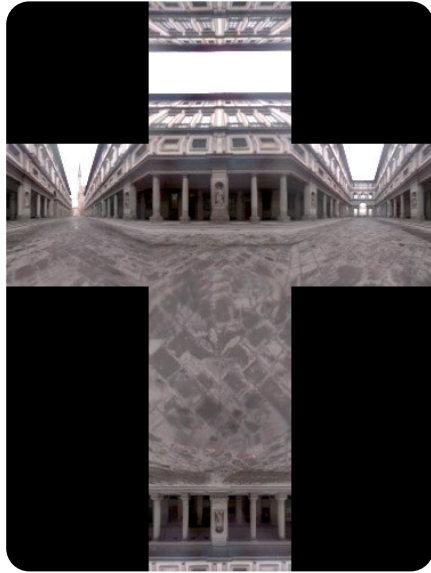


Area light: partly seen

Environment maps: light from everywhere



Mirror ball

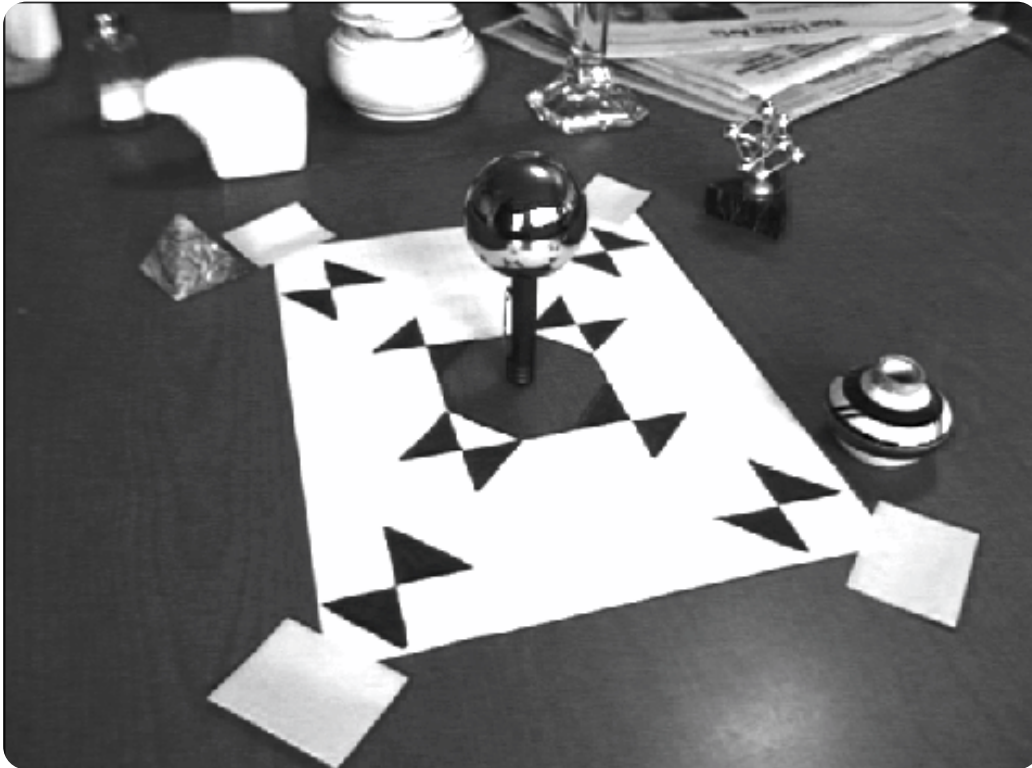


Cube map



Panorama

Capture it with a mirror ball or 360° camera

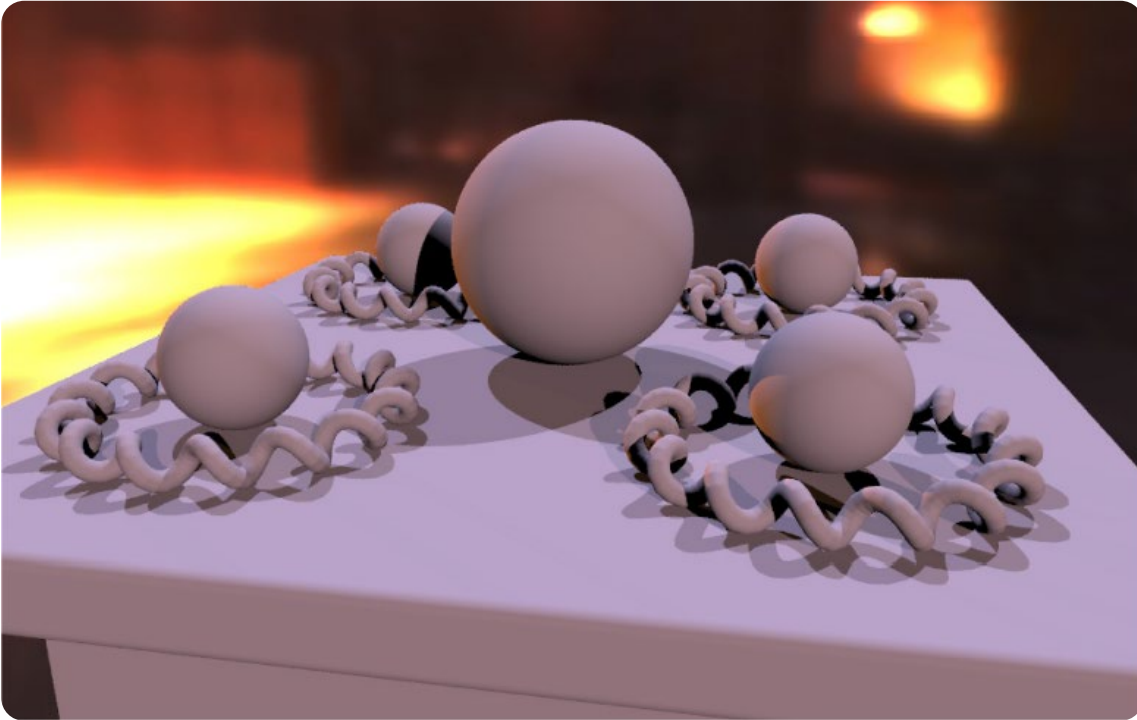


Mirror ball (Debevec, 1998)

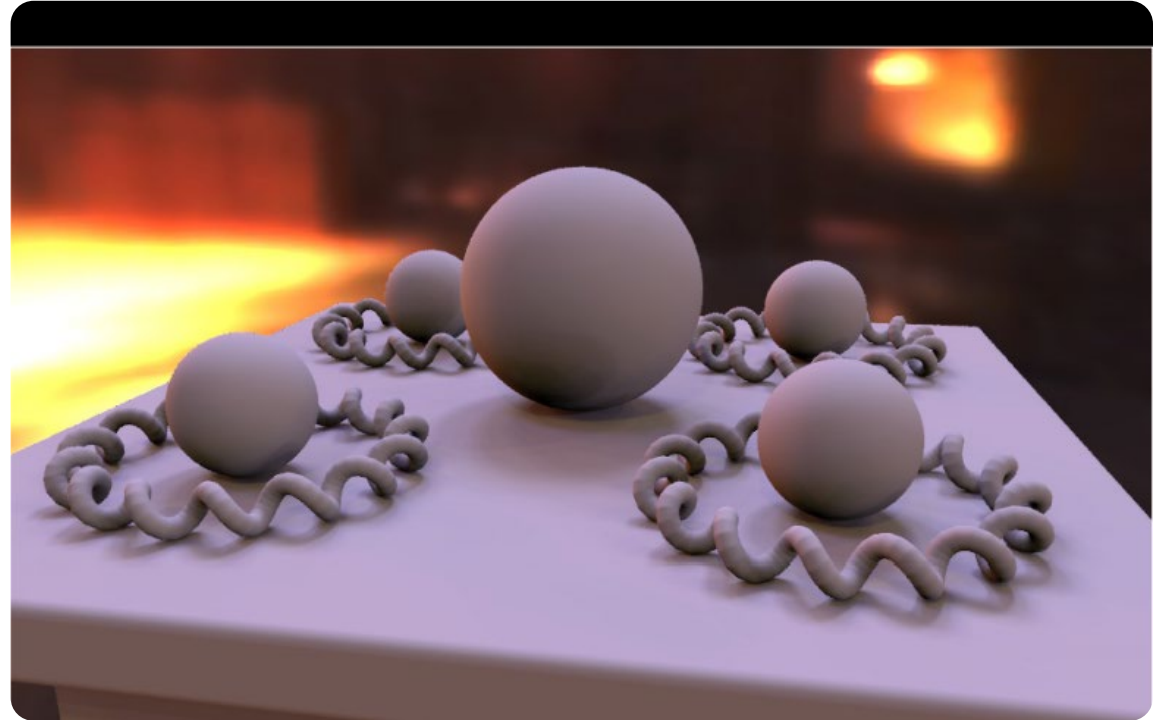


360° camera (RICOH THETA)

Image-based lighting: many small lights



4 lights

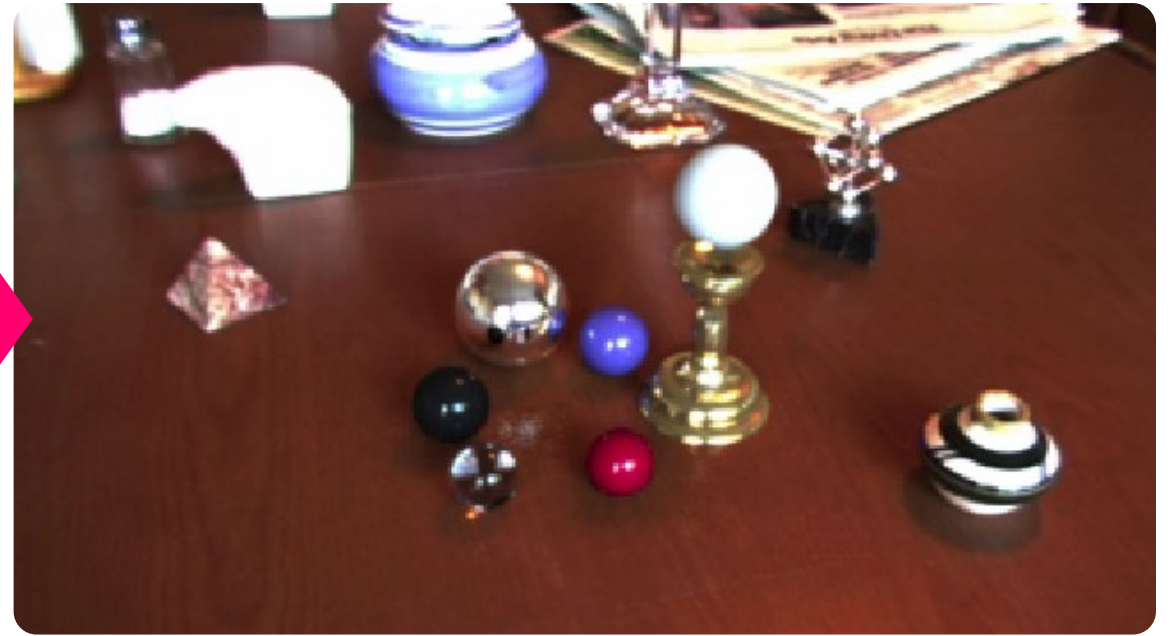


256 lights

Real photo + CG objects, lit the same way

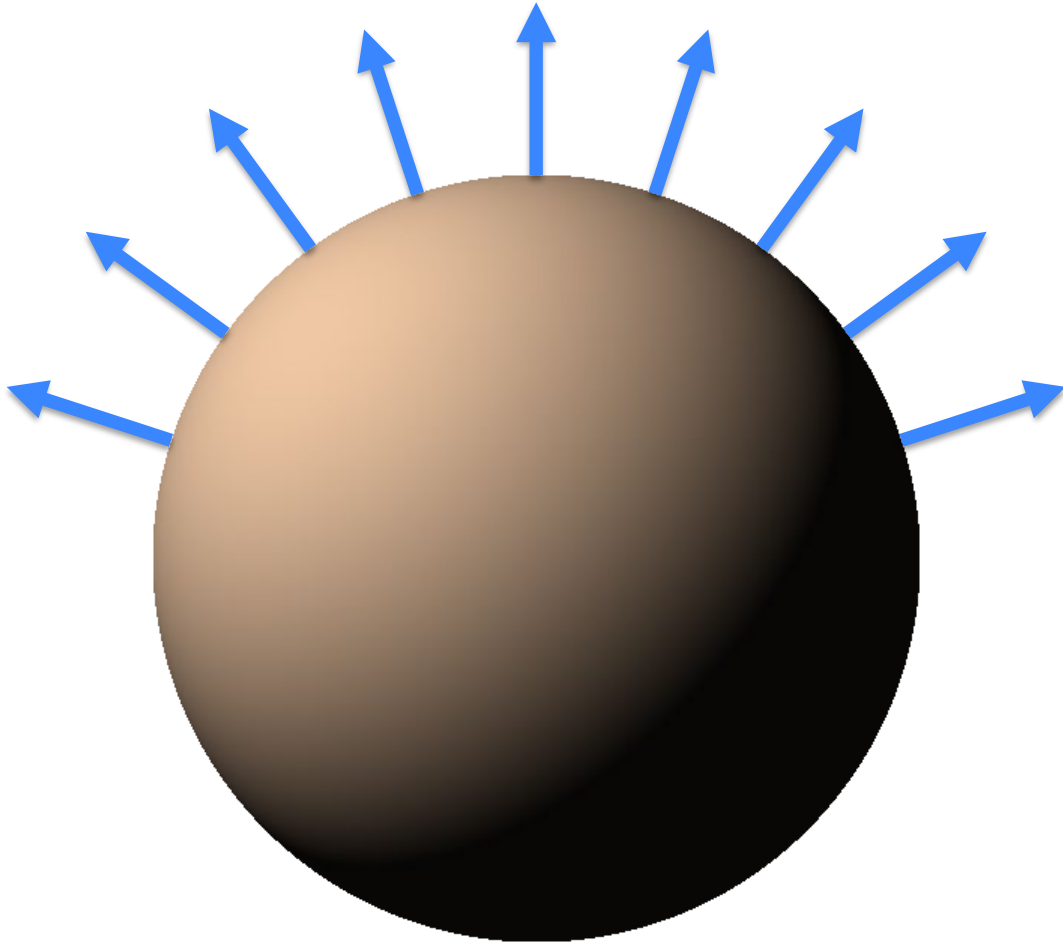


Real photo



With CG objects

Shape: which way does each point face?



- **Normal** = the direction a surface faces
- One arrow for every point
- Shading depends on the normal and the light

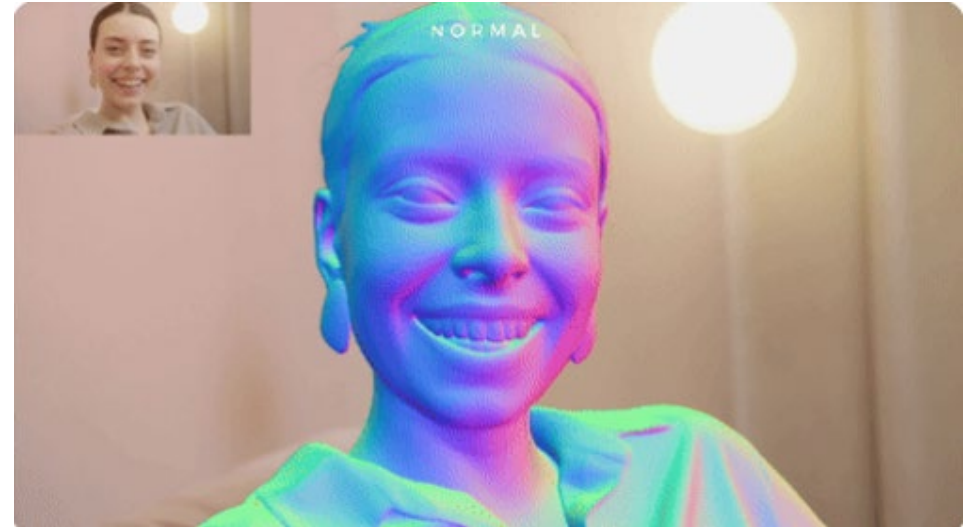
Normal map: directions stored as colors



Red → right

Green → up

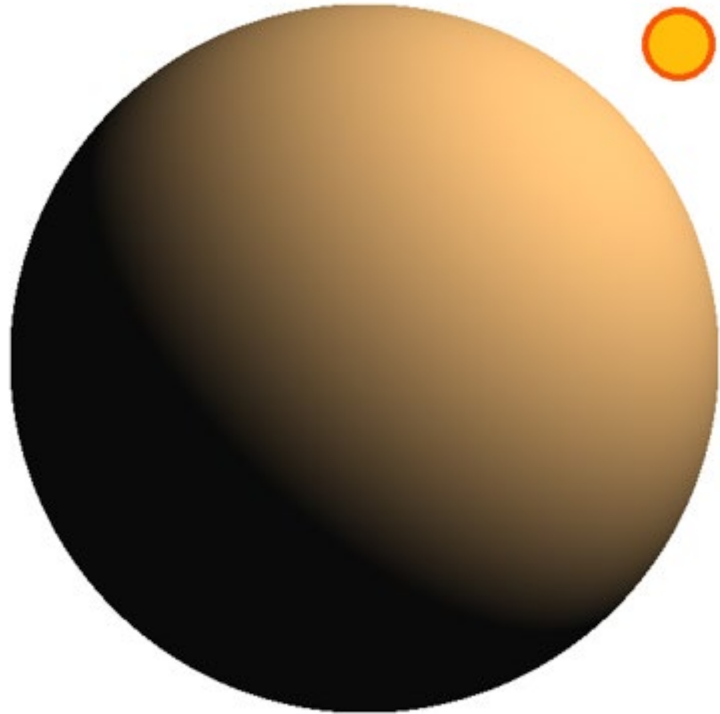
Blue → camera



Sapiens2: photo, then normals

Right: facebookresearch/sapiens2 demo

Facing the light = bright



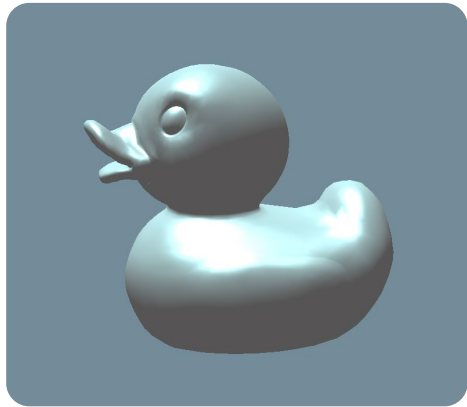
- **Faces the light** → bright
- **Faces away** → dark
- In between → in between
- Light moves → shading follows

Bending normals fakes fine detail



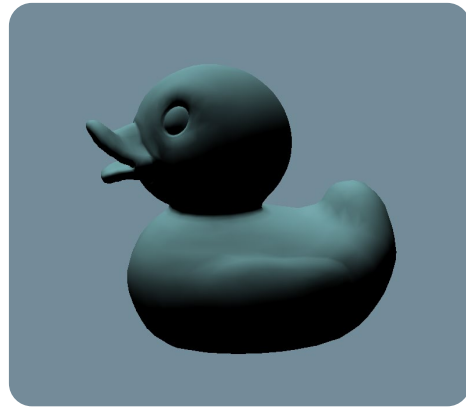
- The surface is smooth
- Only the **normals** are bumpy
- Lighting makes it look detailed
- **Our plan:** normals, no 3D model

Material: matte part + shiny part



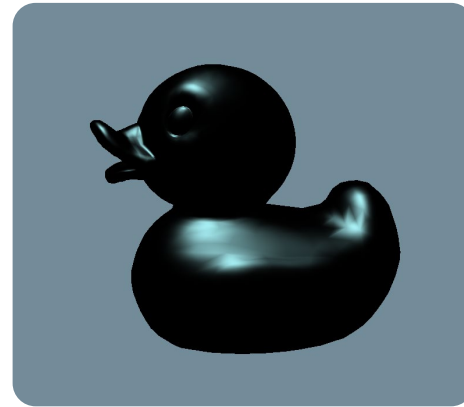
Result

=



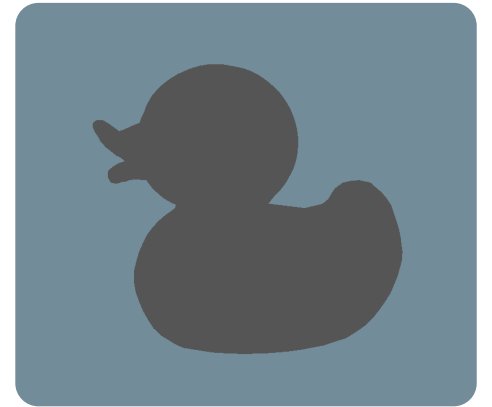
Diffuse (matte)

+



Specular (shiny)

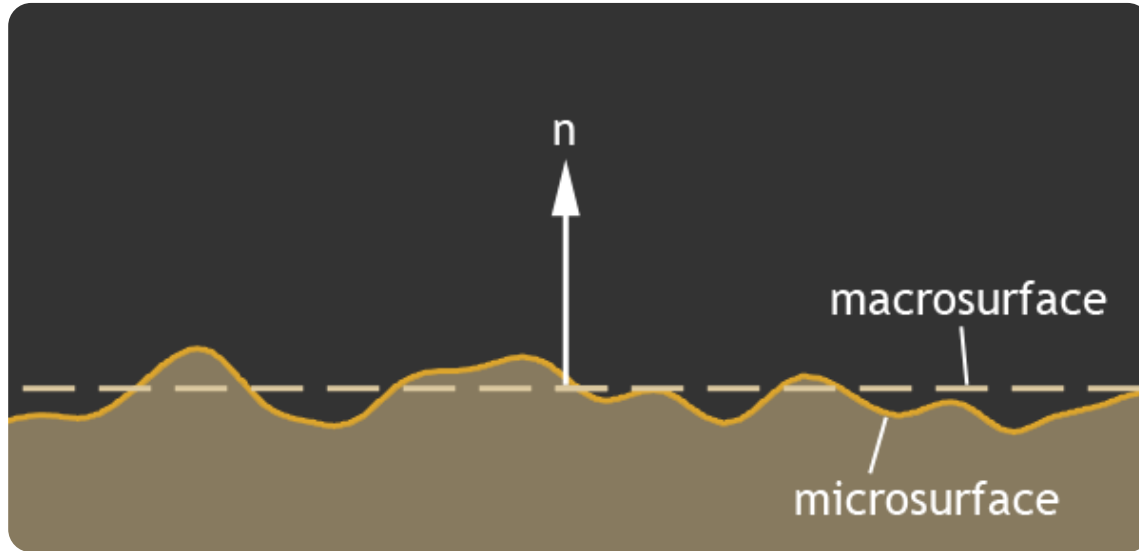
+



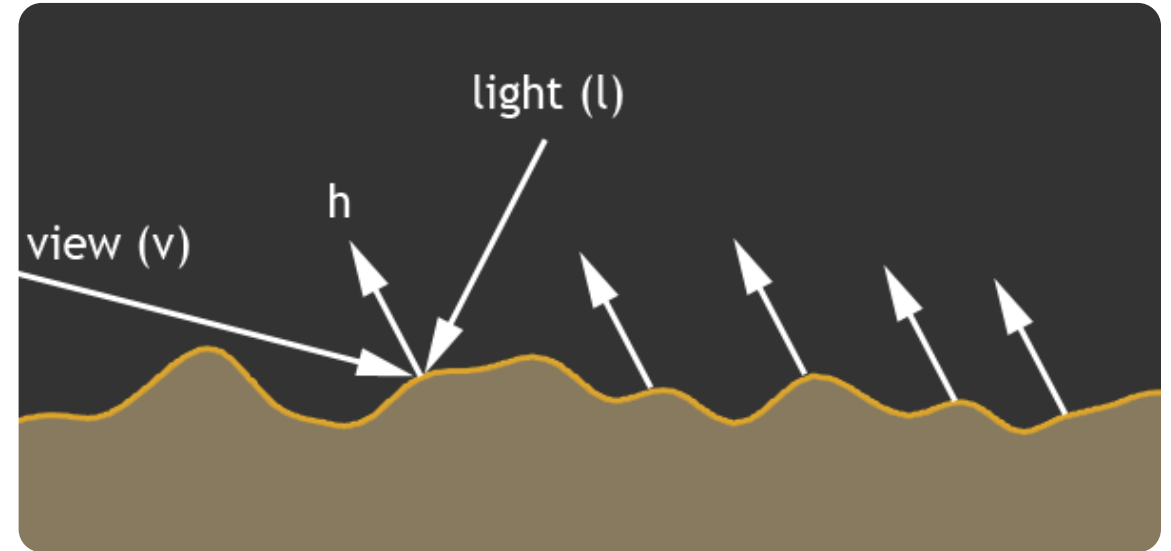
Ambient

This week: diffuse + specular

Shiny or rough: tiny facets decide



Rough on a small scale



Each facet reflects like a mirror

In Unreal Engine: one "Roughness" value

Not every matte surface is simple

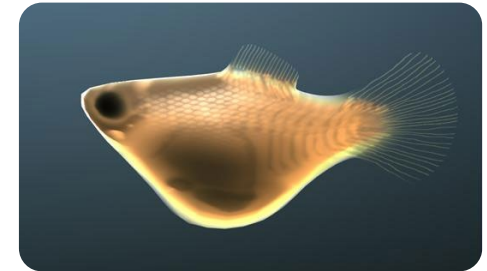


Skin: light travels under the surface



Without

With subsurface scattering



Images: Jensen et al. and Donner & Jensen (subsurface scattering)

Three ingredients make one image



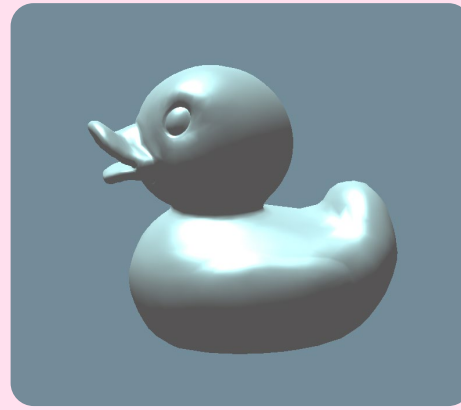
Light

Lamps, sky,
environment
map



Shape

Normals: one
direction per
pixel



Material

Albedo,
shininess,
translucency



Image

4

How film productions do it

Light stages, digital doubles and LED volumes

The Light Stage: a dome full of lights



The dome



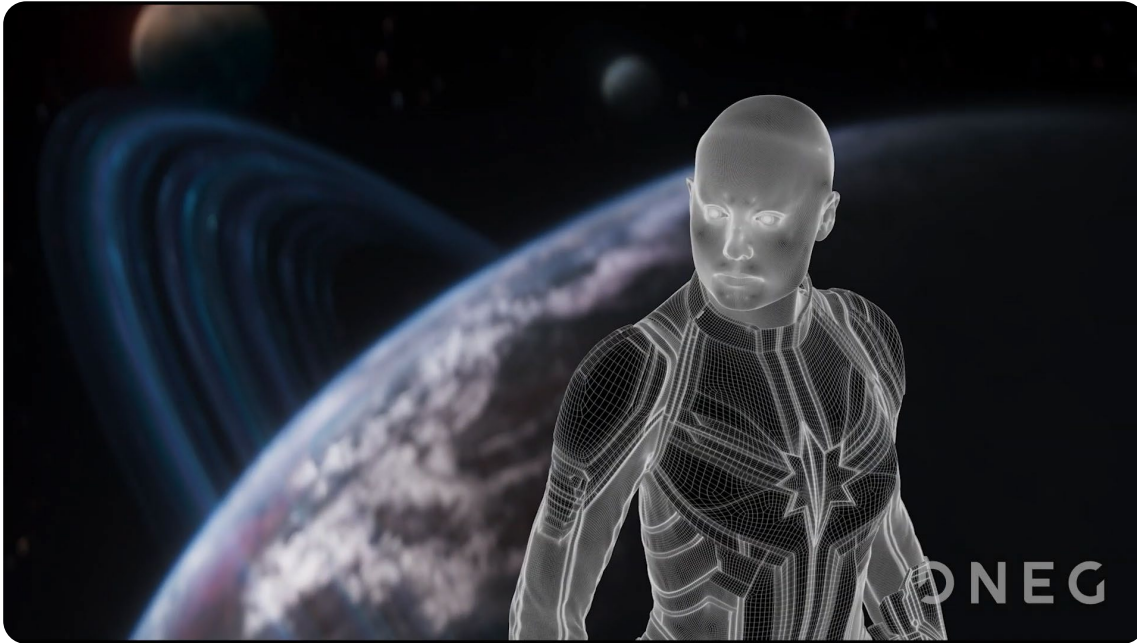
Lights flashing during capture

One light at a time, then mix

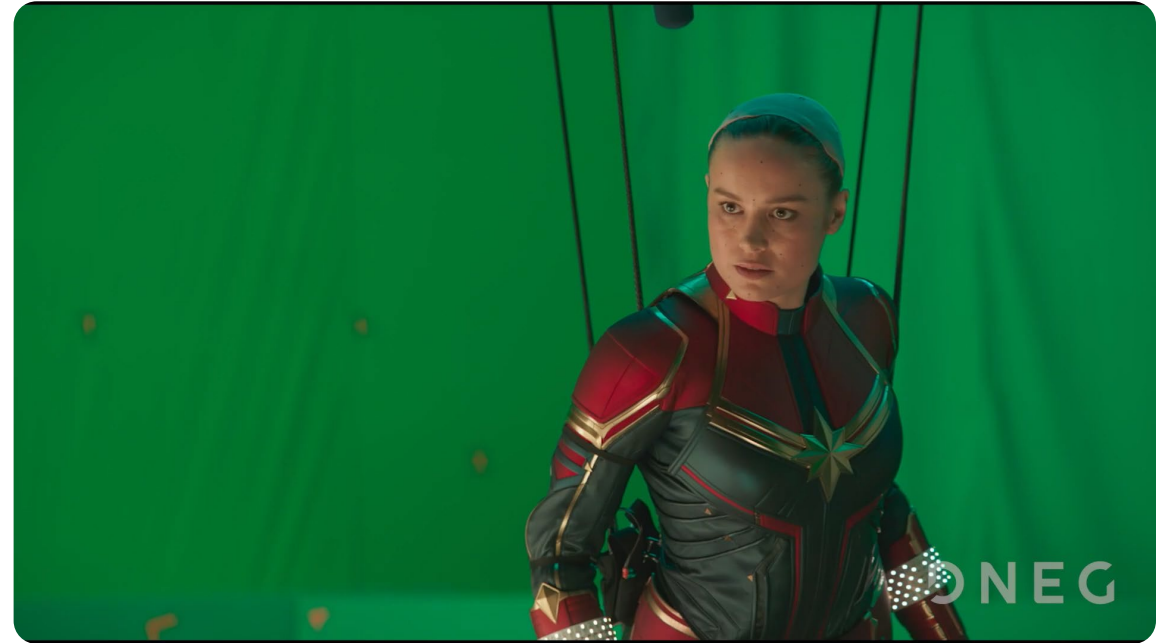


- One photo per light
- Light is **additive**
- Weighted sum = any lighting
- Measured, not guessed
- OLAT: **O**ne **L**ight at **A** Time

Digital doubles are measured, not guessed



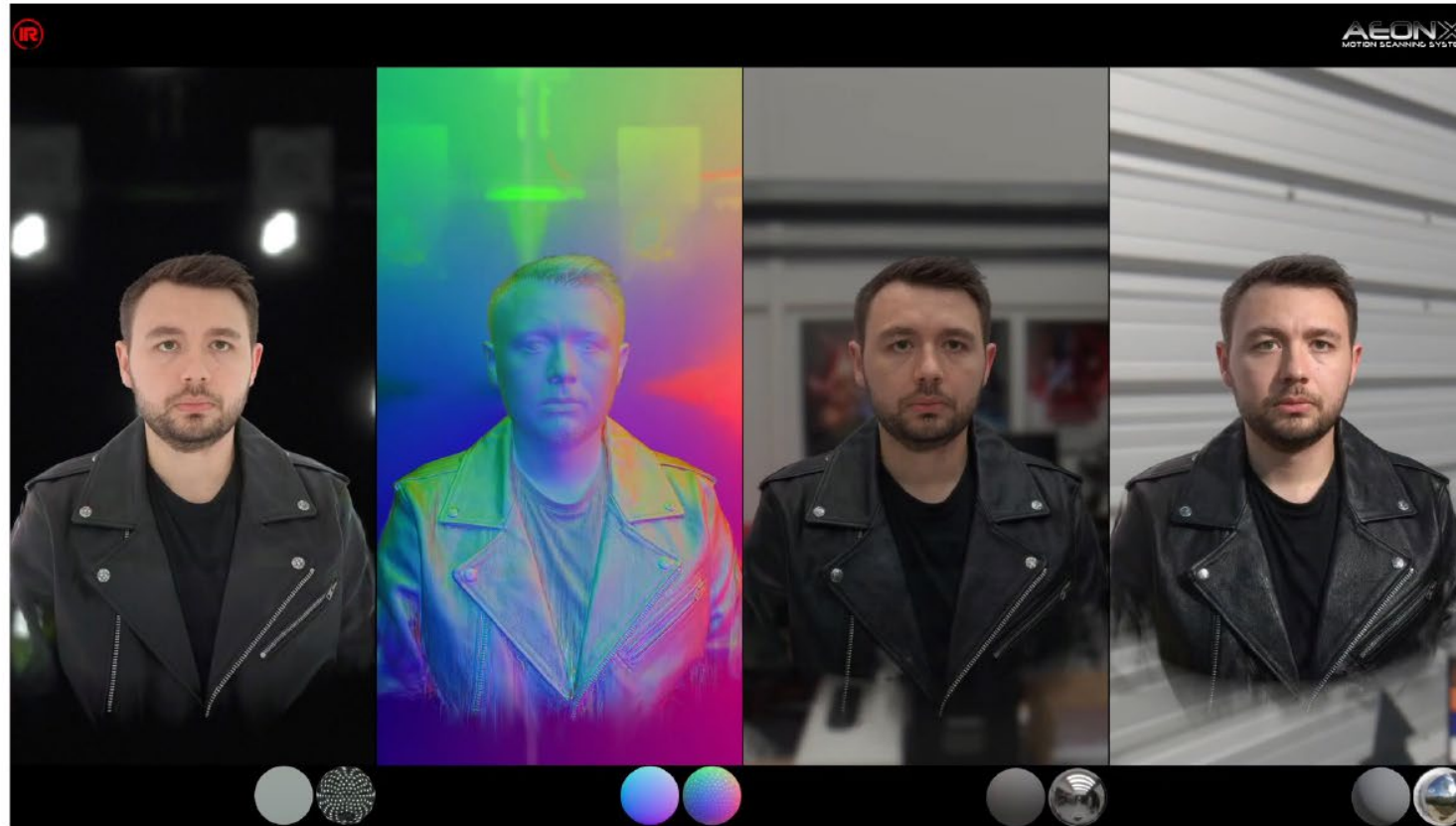
Digital double (CG)



Real footage

Light stages meet Gaussian splats

Gaussian Splatting OLAT captures

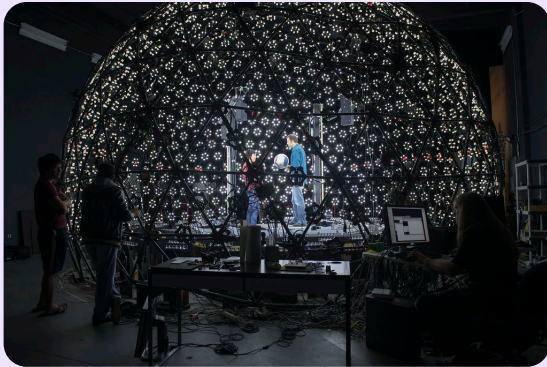


Capture and video from "Infinite Realities"

71

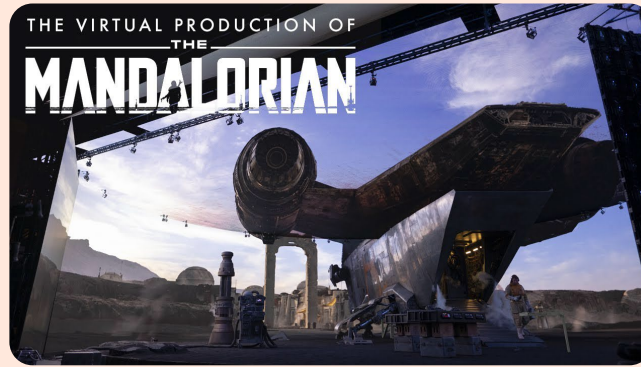
Capture: Infinite Realities · via 3DV 2024 Gaussian Splatting tutorial

Three production solutions compared



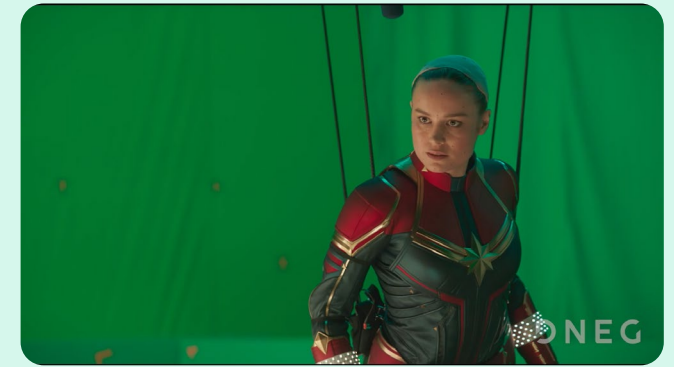
Light stage

- Best quality
- Huge dome
- Not on set



LED volume

- Live, in camera
- Very expensive
- Light fixed



Green screen

- Cheap, flexible
- No useful light
- Needs relighting

5

Relighting with simple capture

No dome, no LED wall: just a camera

What if we only have one camera?



X Light stage



X LED volume



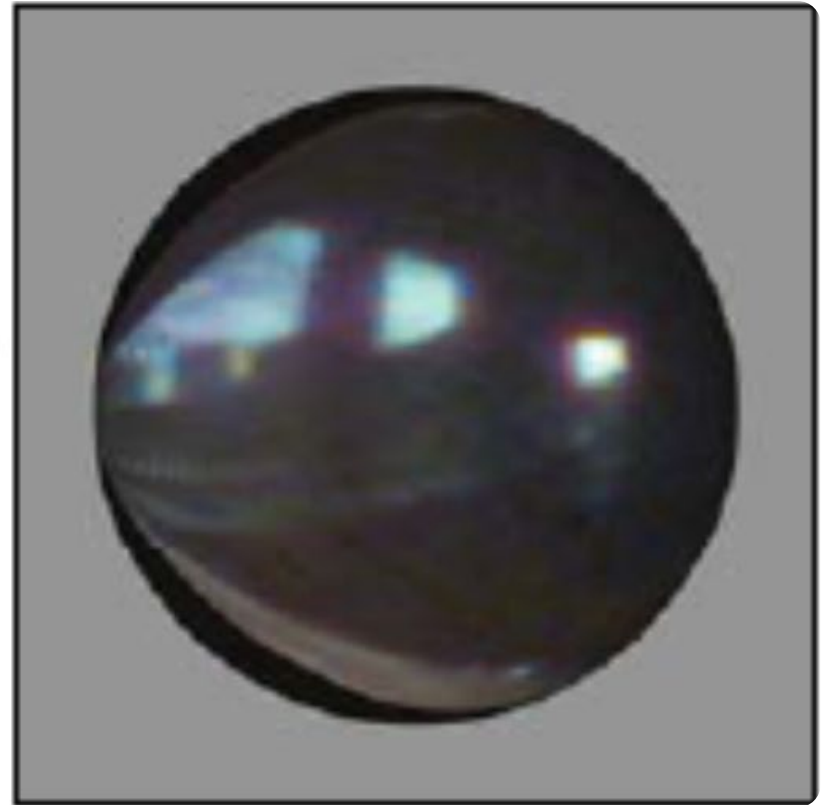
✓ One photo / video

Middle: ILM / Lucasfilm (YouTube thumbnail)

A clever trick: light reflected in the eyes



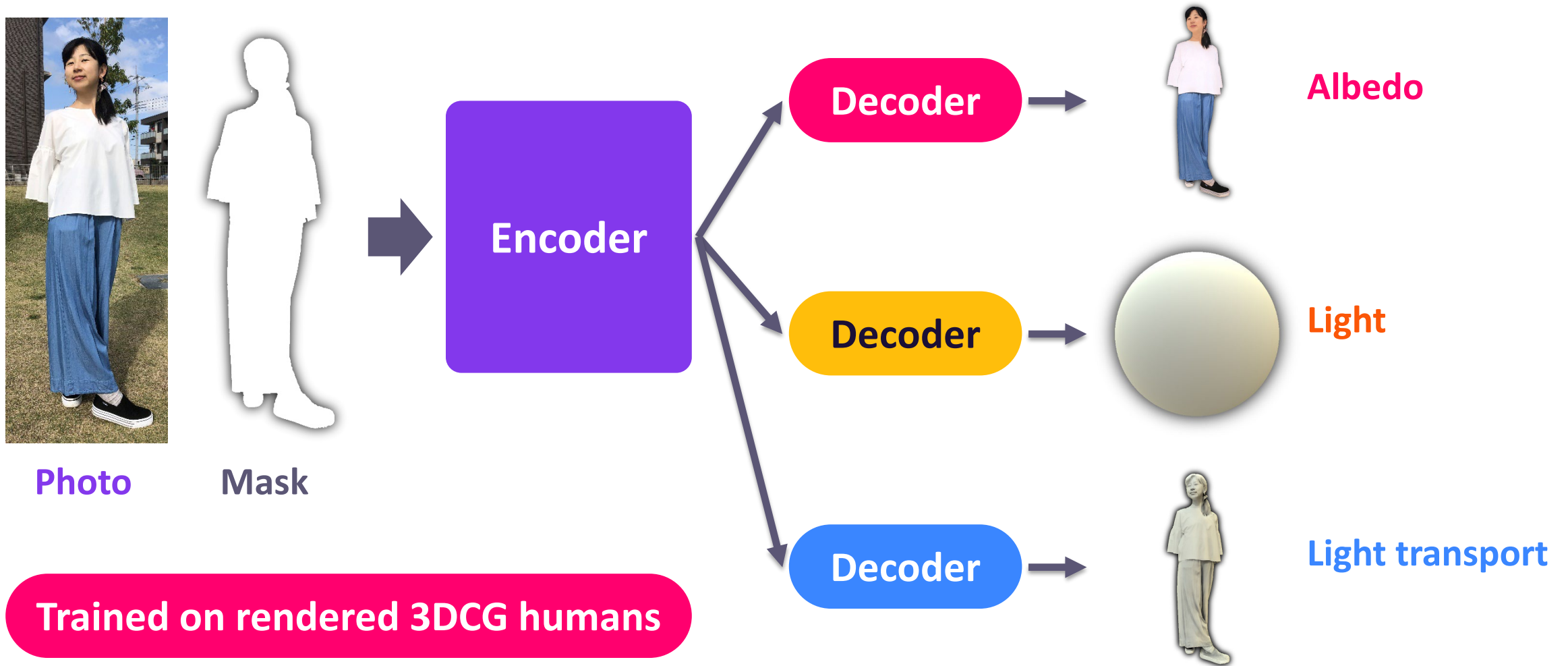
Face photo



Recovered environment

Eye = tiny mirror ball

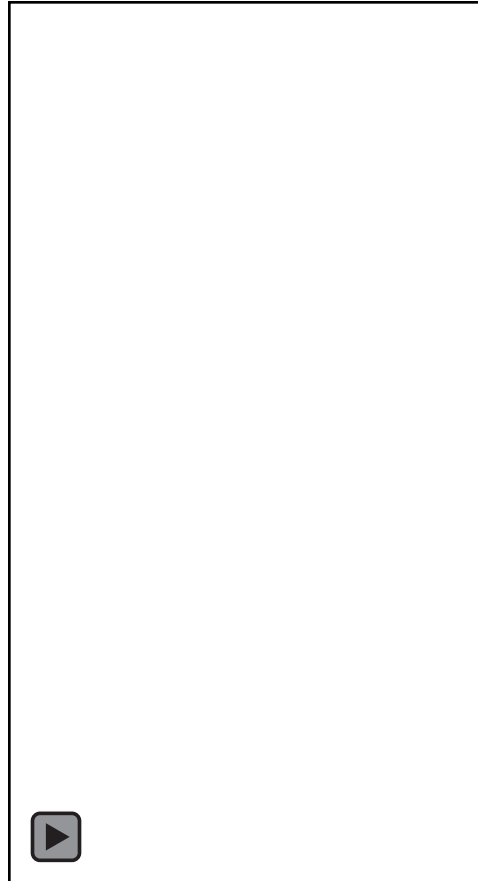
Our 2018 idea: a network decomposes it



Full-body relighting from a single photo



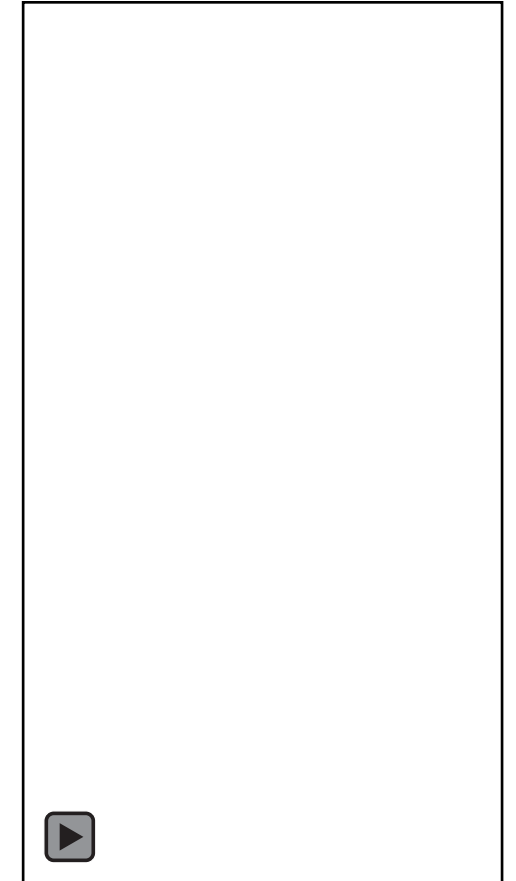
Input



Shading

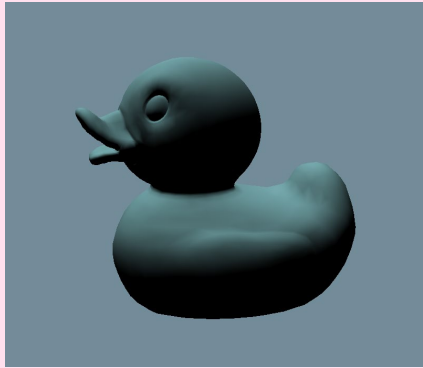


New light



With color

What our 2018 method could not do



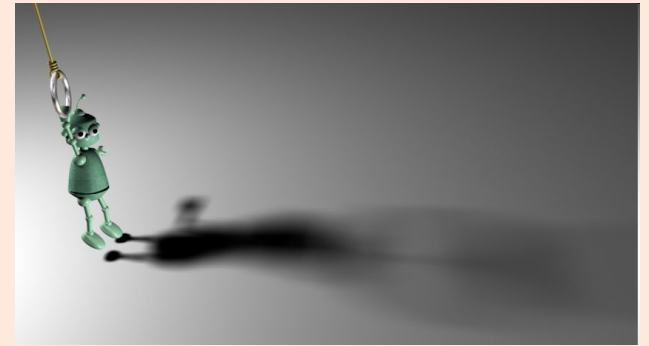
Matte only

No shiny highlights



CG training data

Real photos look
artificial



Soft shadows only

No sharp sun shadows

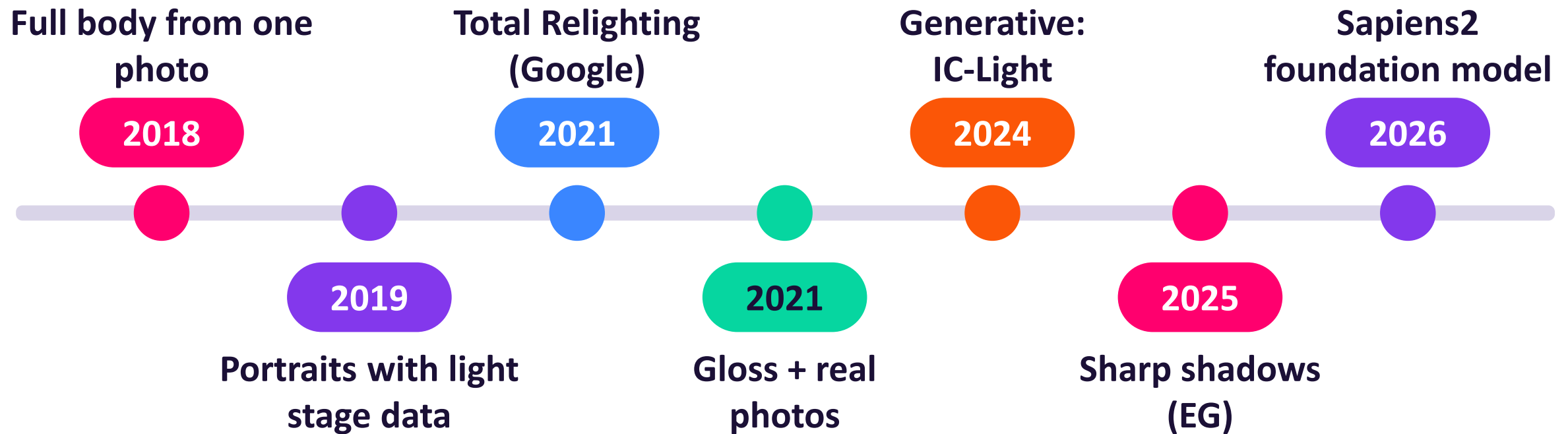
Next: how research solved these three problems

6

Neural relighting: research trends

From light stage data to foundation models

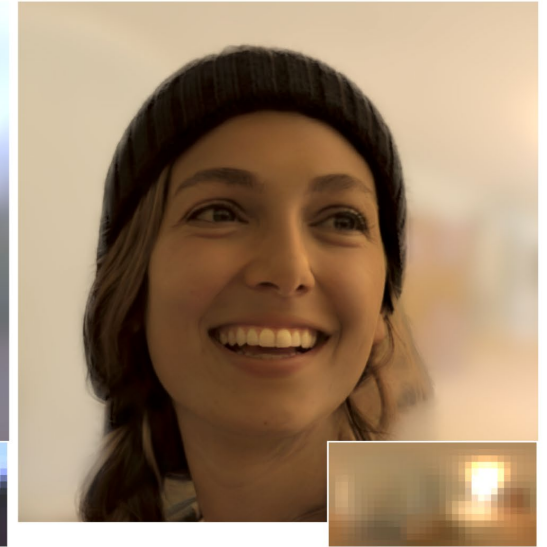
A short timeline of neural relighting



Training on light stage data



(a) Input image and estimated lighting



(b) Rendered images from our method under three novel illuminations



Top: Sun et al. 2019
Bottom: Pandey et al. 2021

Total Relighting (Google, 2021)



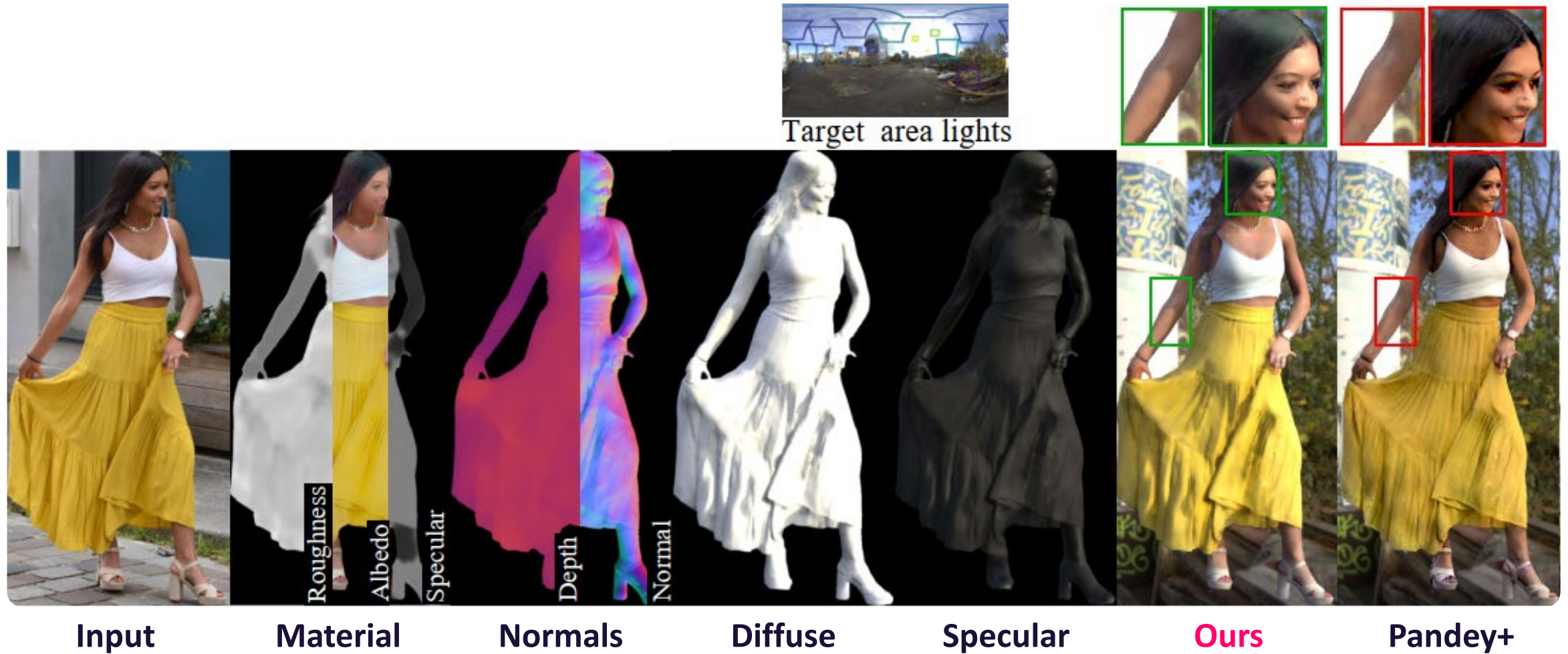
1 Cut out the person

2 Estimate normals + albedo

3 Light from new background

4 Render + composite

Our EG 2025 work: sharp shadows



Key idea: a few area lights + shadow maps



Input



Depth



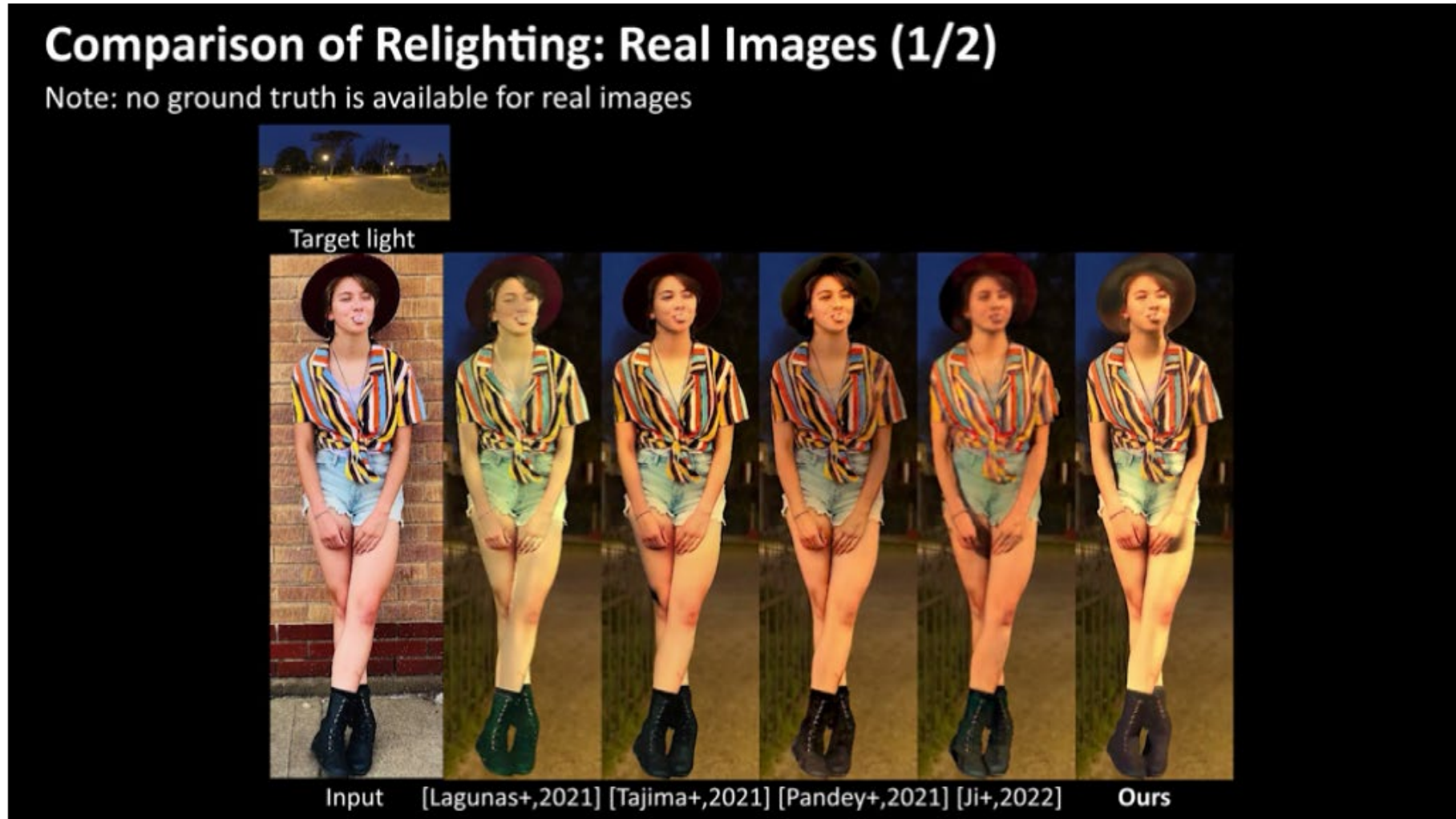
Few area lights



Shadows

Game technique (shadow maps) + a small refinement network

Results on real photos



Two families: physics-based vs generative

Physics-based

- Estimate normals, albedo, ...
- Render with real lights
- Controllable, stable in video

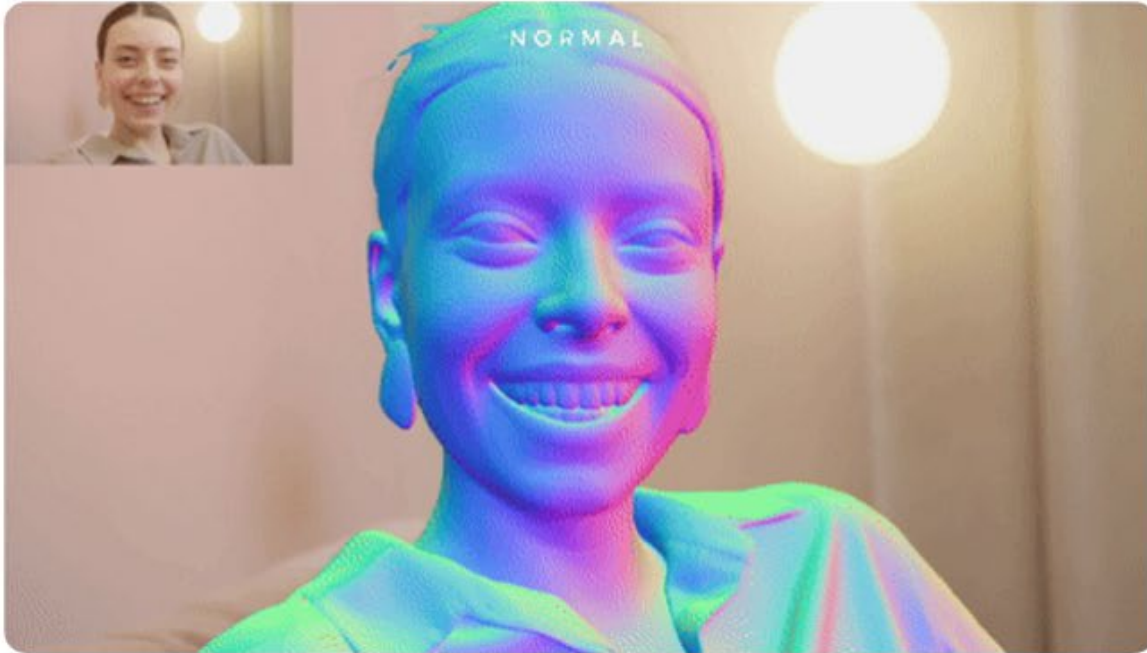
Total Relighting, SwitchLight, ours

Generative

- Diffusion model paints the light
- Prompt or background as input
- Beautiful, but less control, flicker

IC-Light and similar

Foundation models: Sapiens2 (Meta)



- Trained on **~1 billion** human images
- Pose, body parts, depth, normals, matting
- Open code and models

Sapiens2 also handles hard cases



Cluttered background



Fast motion and hands

Still hard: loose hair, glasses, motion blur

Live demo: Sapiens2 in your browser

1 Open the link

2 Upload a photo of a person

3 Look at nose, cheeks, hands

4 Where does it fail?



huggingface.co/spaces/facebook/sapiens2-normal

Click the image to open the demo

7

Our approach this week

Sapiens2 normals + video as albedo + lights in Unreal Engine

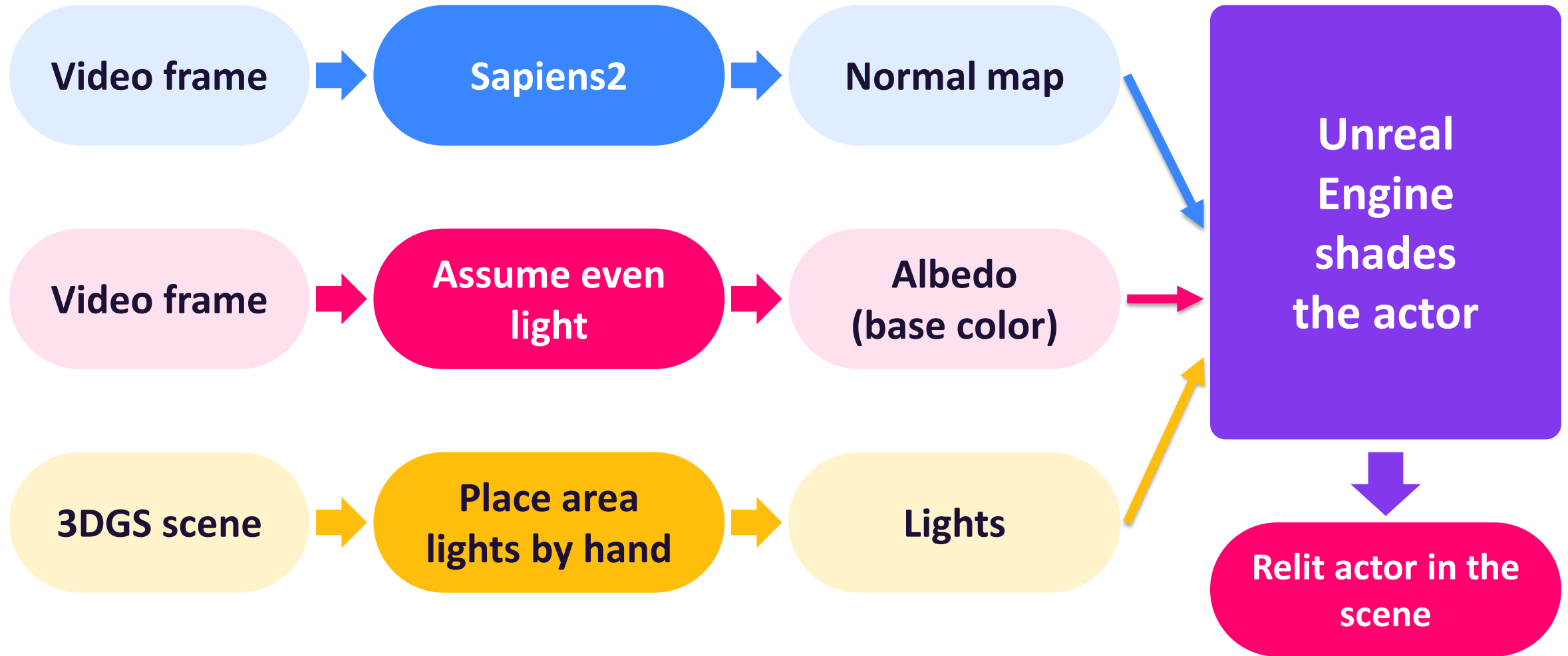
Goal: virtual lights light the actor



- Actor on a green screen
- Virtual scene: 3D Gaussian Splatting
- Virtual lights **shade the actor**
- Simple gear: camera, GPU, Unreal Engine

First test: HSD virtual production group

Our pipeline at a glance



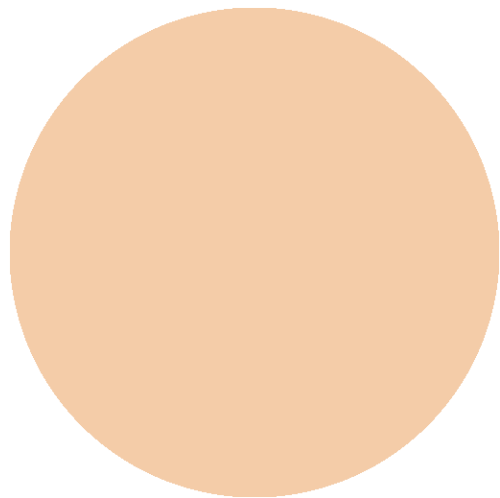
Step 1: normals from video with Sapiens2



- Every frame → one normal map
- Runs on a GPU computer
- Check the **frame rate** early
- Sharp, bright image = better normals

Step 2: the video frame becomes the albedo

- **Assume:** actor lit evenly, no shadows
- Then **frame $\approx$ albedo**
- Unreal multiplies it by the new shading



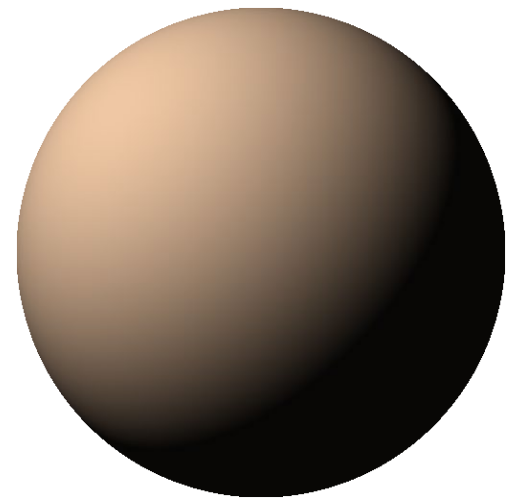
Frame as albedo

×



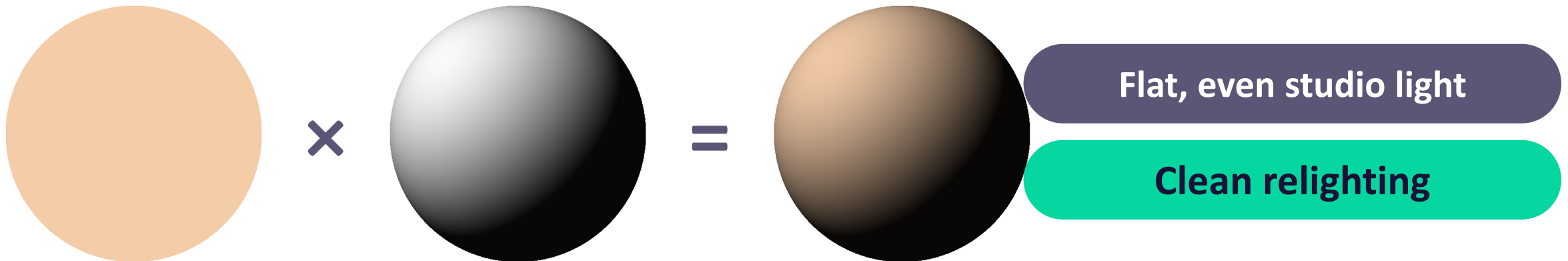
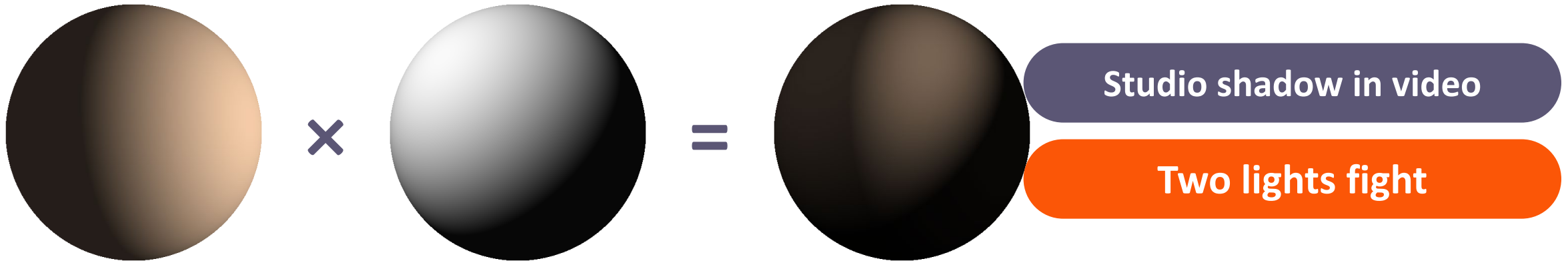
New shading

=



Relit

Why the actor must be lit flat



Step 3: put lights into the 3DGS scene



- Splats look lit, but don't emit light
- Add **area lights** at windows and lamps
- Match color and brightness by eye
- A few lights approximate the room

Image: Volinga plugin for Unreal Engine (YouTube)

Step 4: Unreal Engine does the shading

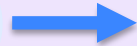
Material of the actor

Video frame



Base Color

Normal map



Normal

Shading model: Default Lit

Not "Unlit": it ignores all lights!



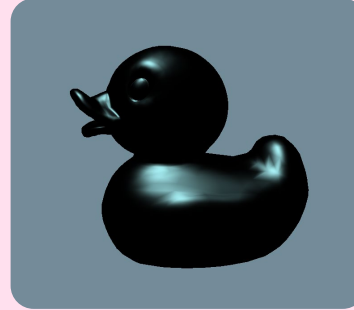
Unreal Engine 5: real-time lighting

What this shortcut cannot do (yet)



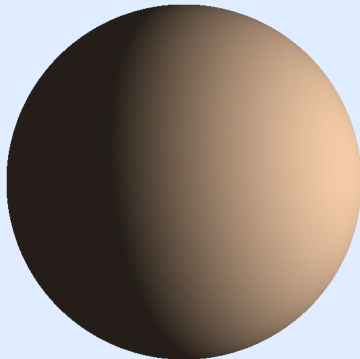
No cast shadows

Normals, not a 3D shape



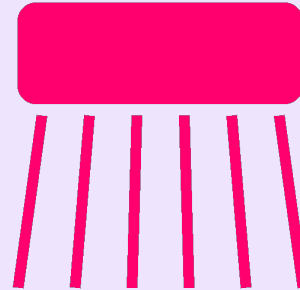
No gloss

Diffuse only: matte look



Baked light stays

Light the actor flat



Splats emit no light

Place lights by hand

From this week to research



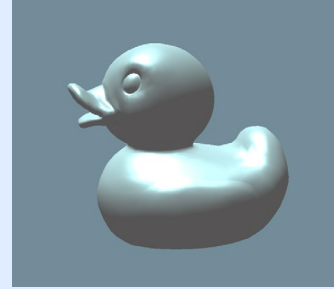
Cast shadows

Depth + few
lights + shadow
maps



True albedo

Remove studio
light with a
network



Gloss

Estimate
roughness and
specular



Stable video

Frame-
consistent video
models

Take-aways

- 1 Green screens give no useful light: relighting adds it in software
- 2 $\text{Image} = \text{light} \times \text{shape} \times \text{material}$: take apart, swap the light, rebuild
- 3 This week: Sapiens2 normals + video as albedo + lights in Unreal

Test: move a light across the face. If the nose behaves, it works!

Thank you!

Questions?

Sapiens2 code and demos: github.com/facebookresearch/sapiens2

Sapiens2 normal demo: huggingface.co/spaces/facebook/sapiens2-normal

Total Relighting: augmentedperception.github.io/total_relighting

Poly Haven HDR environment maps: polyhaven.com/hdris

References

Kanamori & Endo, Relighting Humans, SIGGRAPH Asia 2018
Adelson & Pentland, The perception of shading and reflectance, 1996
Debevec, Rendering Synthetic Objects into Real Scenes, SIGGRAPH 1998
Debevec et al., The Light Stage (USC ICT)
Nishino & Nayar, Eyes for Relighting, SIGGRAPH 2004
Sun et al., Single Image Portrait Relighting, SIGGRAPH 2019
Pandey et al., Total Relighting, SIGGRAPH 2021
Tajima et al., Relighting Humans in the Wild, Pacific Graphics 2021
Zhang et al., IC-Light, ICLR 2025
Kanamori lab, all-frequency shadows for human relighting, Eurographics 2025
Meta, Sapiens2, ICLR 2026