Diplomarbeitspräsentationen der Fakultät für Informatik

Accurate Soft Shadows in Real-Time Applications

Masterstudium:
Computergraphik &
Digitale Bildverarbeitung

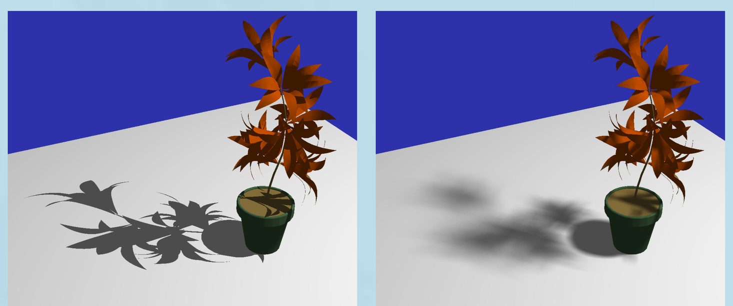
Michael Schwärzler

Technische Universität Wien
Institut für Computergraphik und Algorithmen
Arbeitsbereich: Computergraphik
Betreuer: Associate Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Michael Wimmer
Betreuender Assistent: Univ.Ass. Dipl.-Ing. Mag.rer.soc.oec. Daniel Scherzer

1

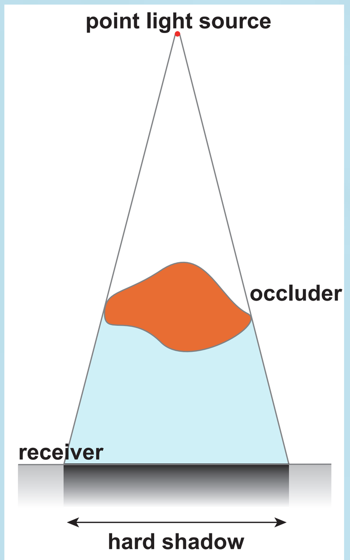
In this thesis, the generation and use of **soft shadows in real-time rendering** is discussed:

- Hard shadow algorithms are already widely used in games & applications
- The **Fast & correct** calculation of soft shadows is a complex task & area of active research!
- Soft shadows **significantly increase the realism of the generated images**:



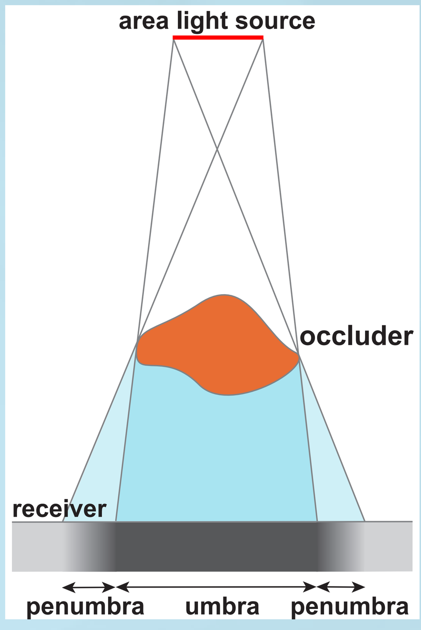
We present a **new algorithm**, which is capable of rendering **physically accurate soft shadows in real-time** by exploiting the **temporal coherence** between neighboring frames.

2



In case of a **point light source** (hard shadow), shadowing is a **binary decision**:

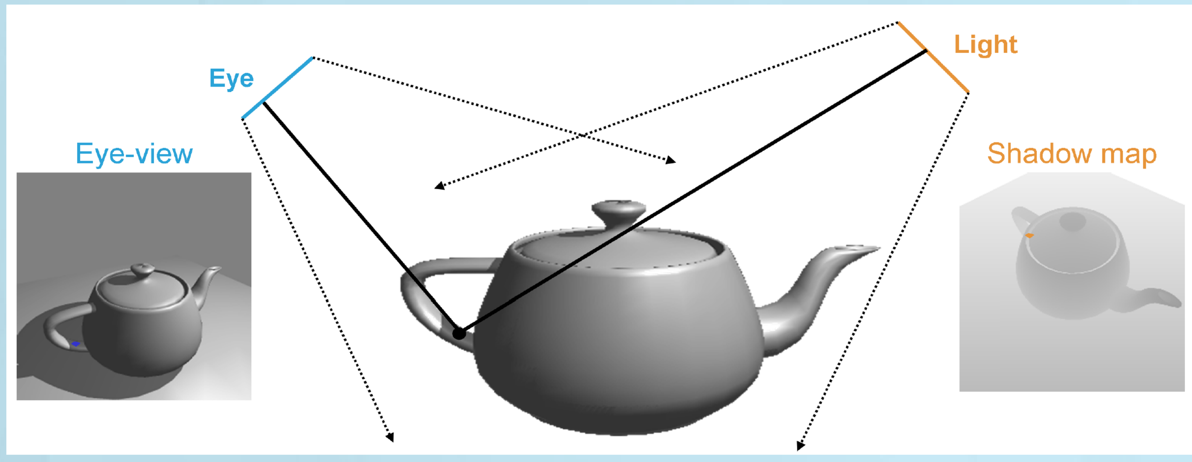
- If the light source is **visible from the point's position**, it is **illuminated**
- **Otherwise**, the point is in **shadow**



For **area light sources**, things are more complicated: If the light source is **partly visible** from the position of the affected point, it is neither completely lit, nor completely shadowed, but in the **penumbra**:

- Soft shadow = union of umbra & penumbra
- The exact calculation of umbra and penumbra is complex: It implicates solving a **three-dimensional visibility problem**
- Umbra & penumbra extents depend on the relationship between **light source size**, distance from occluder to receiver and distance from light source to occluder.

3



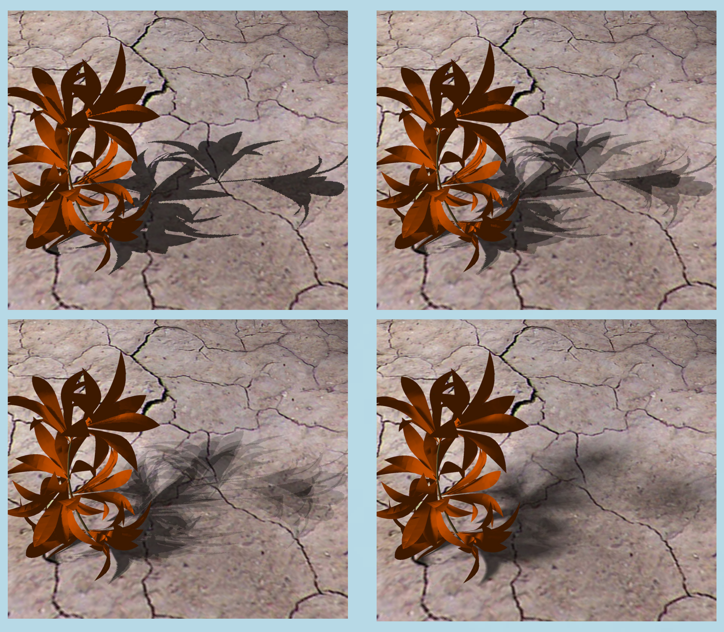
Our algorithm is based on the widely-used **shadow-mapping algorithm**, which generates **hard shadows** from a point light source:

- First, the scene is viewed from the **position of the light source**
- The depth values of the fragments are stored in a **shadow map (SM)**
- The shadow map has to be updated whenever a **movement** occurs

In the second pass, the scene is rendered from the **camera position**:

- Every fragment is **transformed into light space**
- Its distance to the light is **compared** to the corresponding **value in the shadow map**
- If the **distance** to the current fragment is **larger** than the shadow map value, it lies in **shadow**; otherwise it has to be **illuminated**

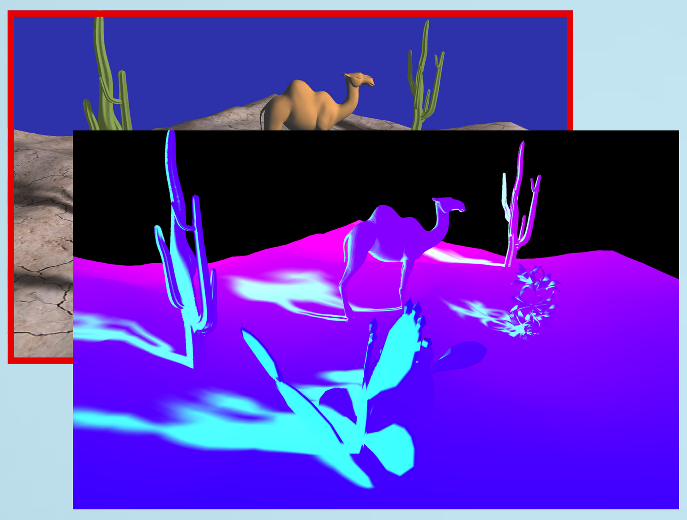
4

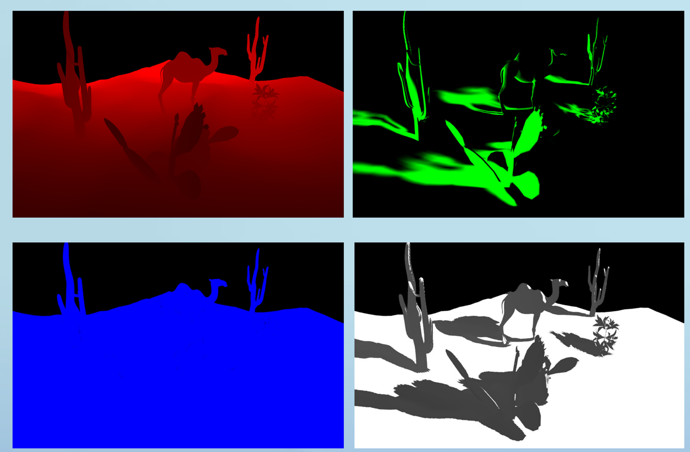


The **soft shadow information** is calculated by approximating the area light source by ***n* different point light sources:**

- Samples **randomly distributed** over the area light
- **Each frame**, a shadow map is created **for one single point light source**
- **Sum** of hard shadows ➔ **soft shadow**

5





The calculation of hundreds of shadow maps per frame is very costly and makes real-time frame rates nearly impossible

➔ Only a **single shadow map** is evaluated per frame!

- exploit **temporal coherence** between frames
- store the shadowing information in a **screen-space *shadow buffer*** - a second render target texture with 4 channels, in which the shadow data can be saved.

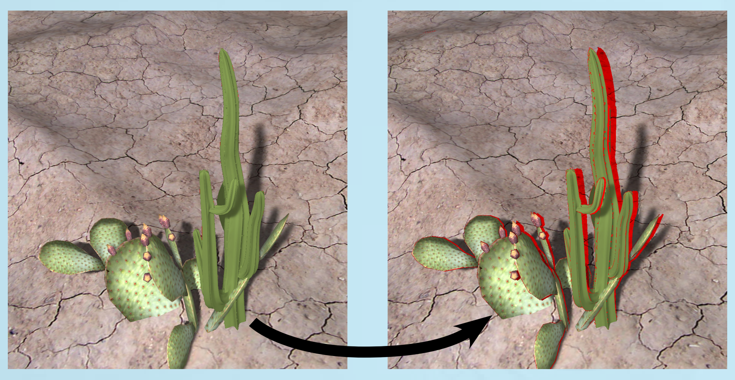
Fragments are **reprojected** (to account for camera movement) **from the new frame into the old**:

- Their **depths** values are **compared** to the stored depths
- If the depth difference is **smaller than a predefined threshold,** the new and the old fragment are **considered equal**
- ➔ Fragment data from the previous frame is **reused!**

Upper left image: **visualization of the shadow buffer**:

- Red channel: **depth**
- Green Channel: **number successful shadow tests**
- Blue channel: **number of all shadow tests**
- α-channel: **penumbra size estimation**

6



Camera movement ➔ disocclusions

- Detected in **depth comparison** step (**red**)
- **No shadowing information from previous frames** in the shadow buffer for them!

➔ We sample the **neighboring fragments** in the shadow buffer, use these values to generate an **estimate**, and **blend** it together with the correct solution:

- Sampling **kernel size** depends on penumbra size
- estimated in a **blocker search** in the SM
- stored in **α-channel** ➔ reuse in next frame(s)

This image shows a scene with overlapping occluders consisting of 70k triangles rendered with our new method at 344 frames per second.

Our results show that this new approach is **as fast as the fastest single sample approaches**, but at the same time provides **significantly better image quality**: Apart from being **physically accurate**, our method **does not suffer from typical single-sample artifacts**.

Kontakt: michael@schwaerzler.com