

Sample Distribution Shadow Maps

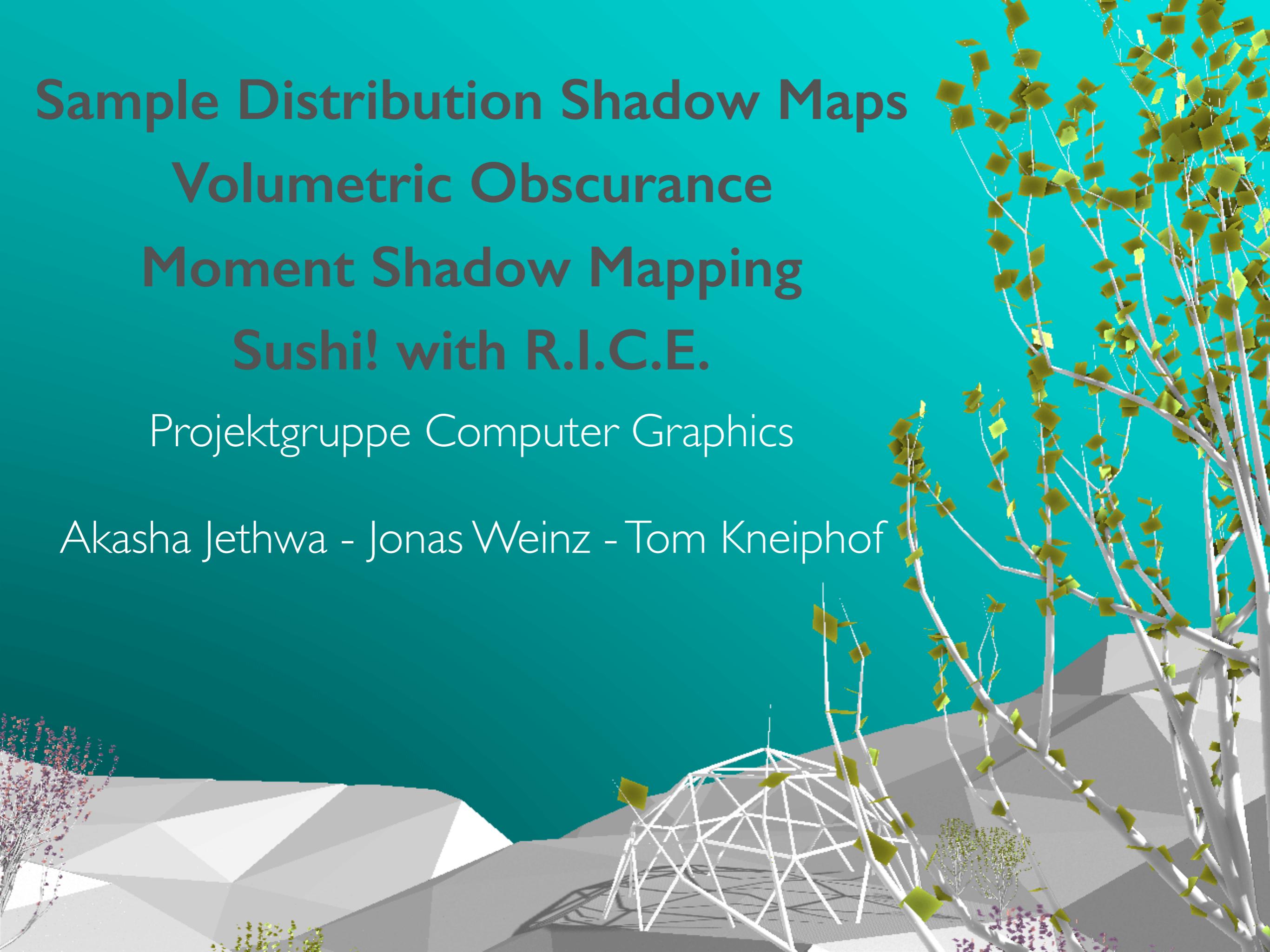
Volumetric Obscurrence

Moment Shadow Mapping

Sushi! with R.I.C.E.

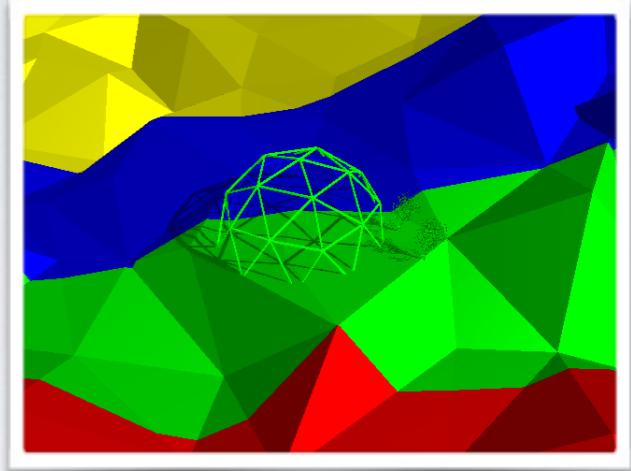
Projektgruppe Computer Graphics

Akasha Jethwa - Jonas Weinz - Tom Kneiphof

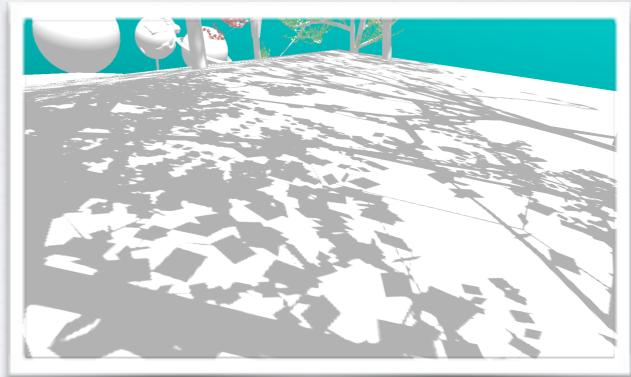


I. INHALT

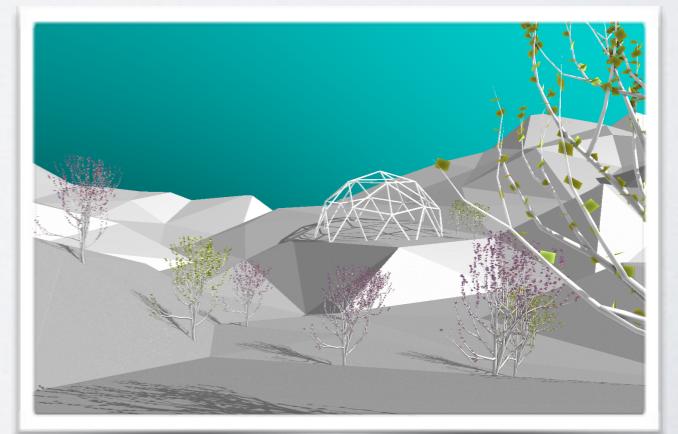
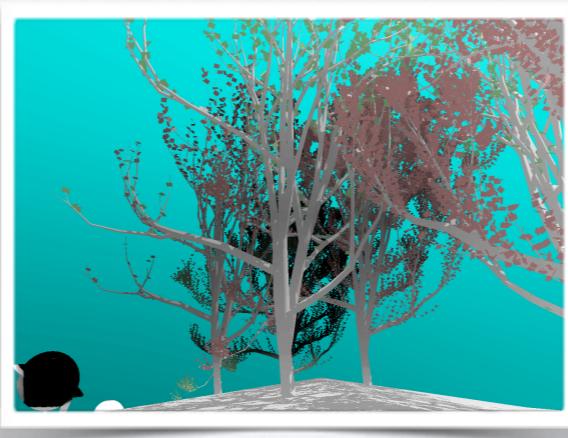
2. Sample Distribution Shadow Maps



3. Volumetric Obscuration



4. Moment Shadow Mapping

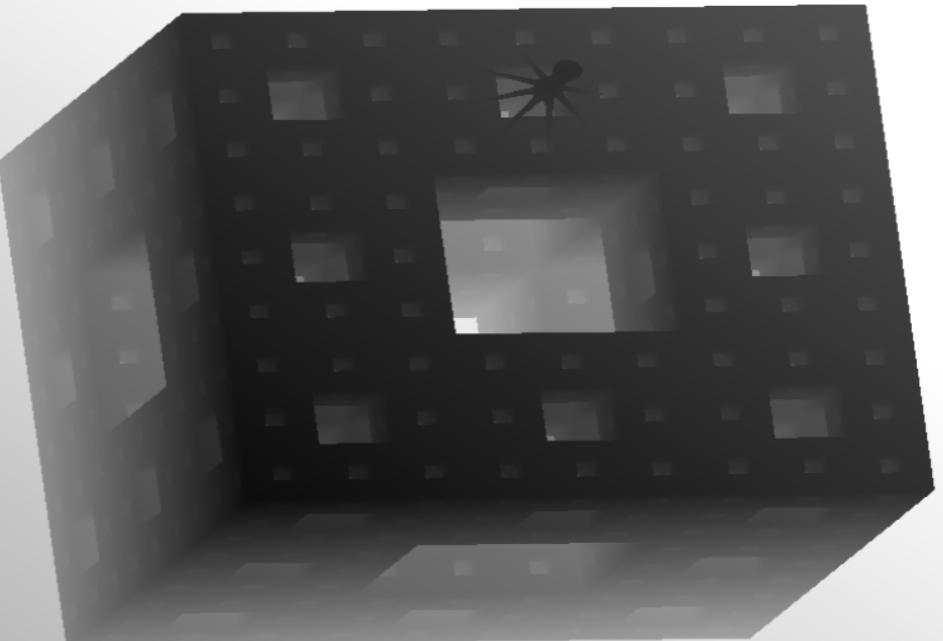


5. Sushi! with R.I.C.E.

6. Fazit und Aussichten

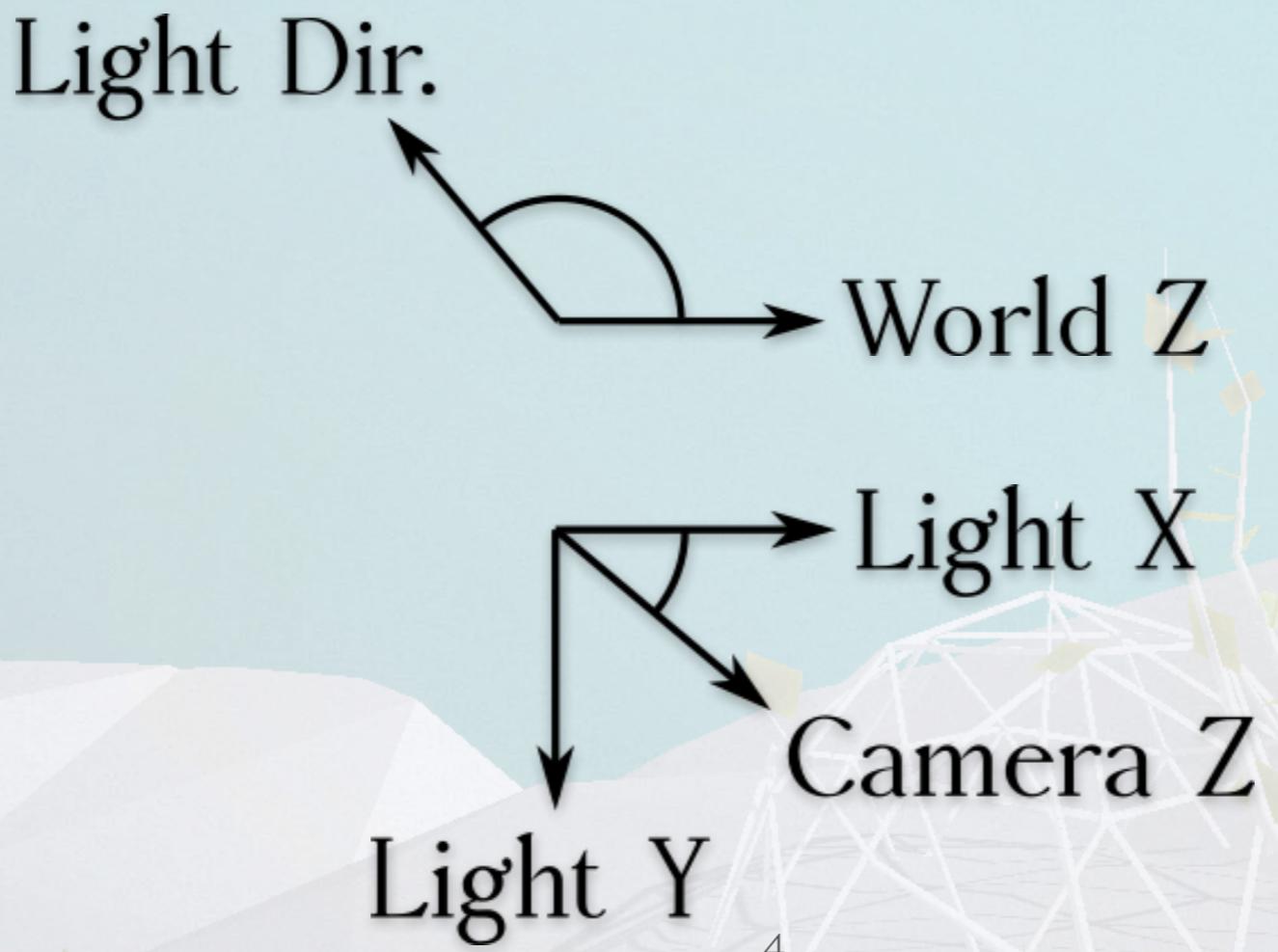
2. I SDSM: MOTIVATION

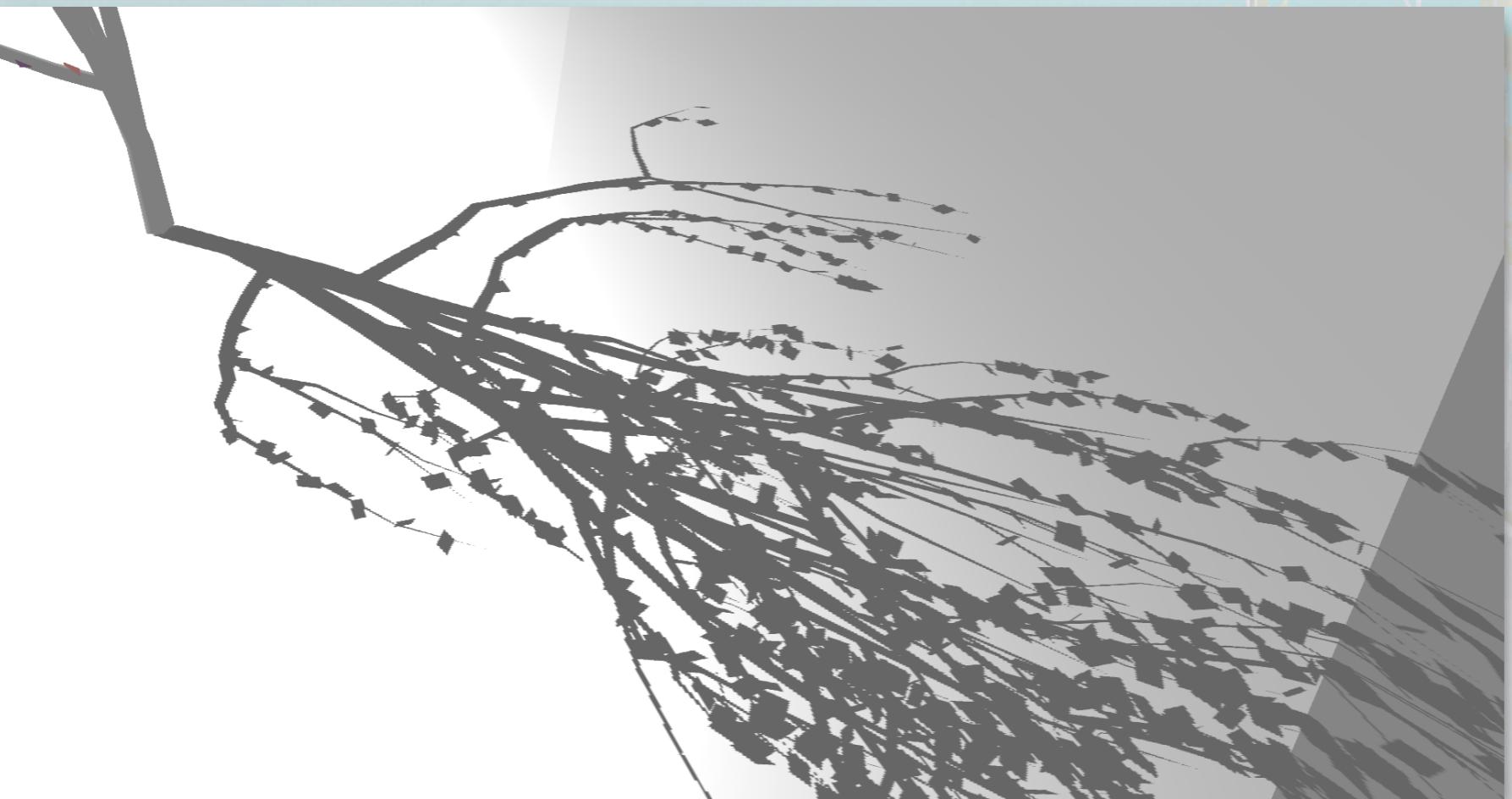
- Echtzeit Schatten
- Punktlichtquellen
- RenderTiefe aus Sicht des Lichtes



2.2 SDSM: LIGHT SPACE

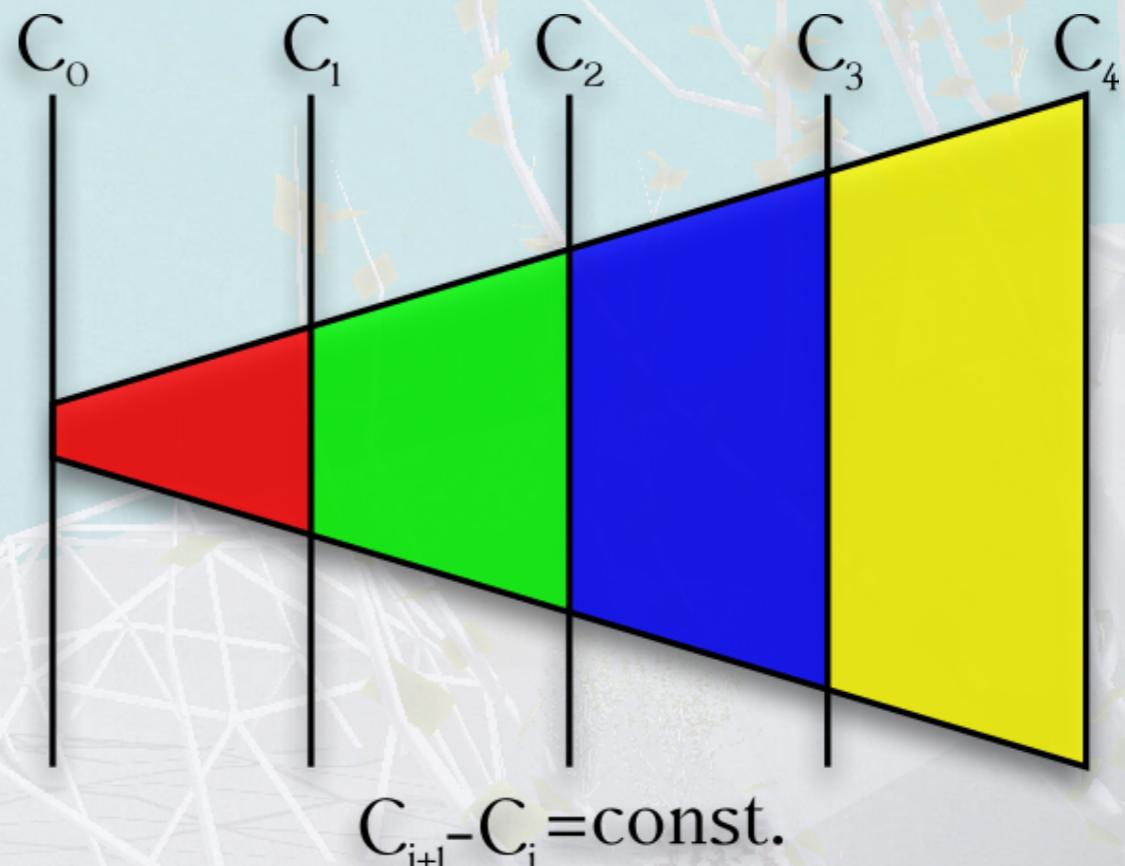
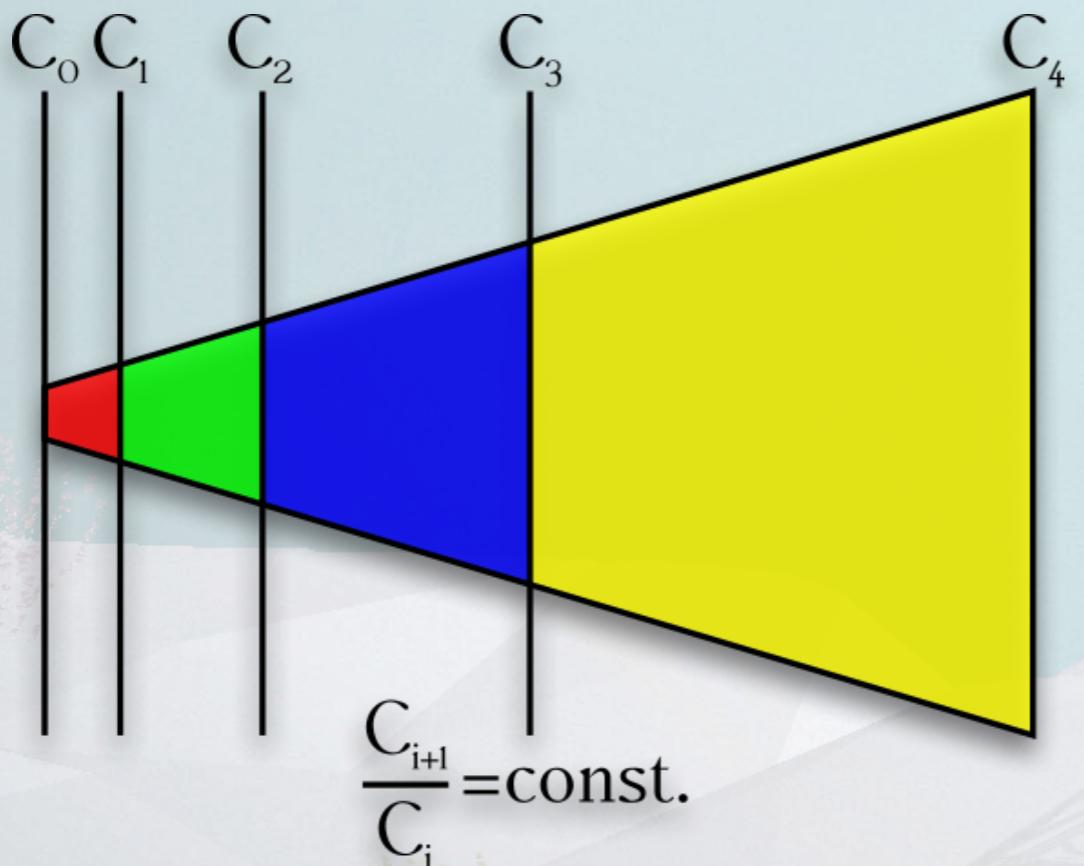
- Rotiere Light Direction auf Z-Achse
- Rotiere Kamera Z-Achse um die resultierende Z-Achse auf X-Achse





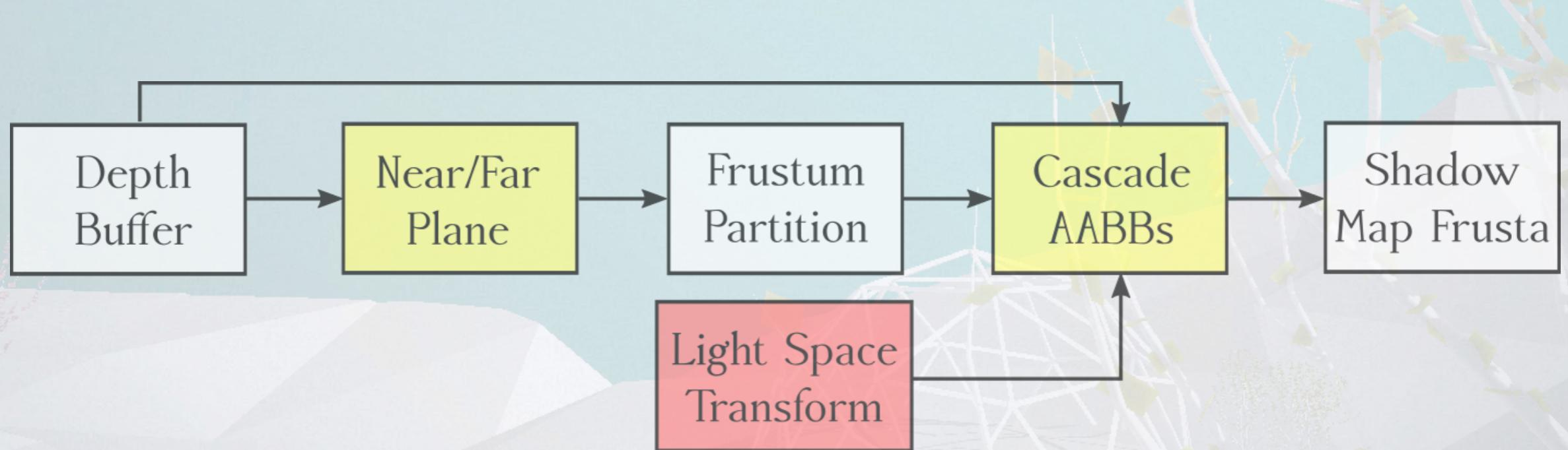
2.3 SDSM: PARALLEL SPLIT SHADOW MAPS

- Partitioniere Kamerafrustum entlang Z-Achse
- Eigene Shadowmap pro Kaskade
- Vermeide Oversampling, Undersampling, Geometrisches Aliasing
 - logarithmisch
 - uniform



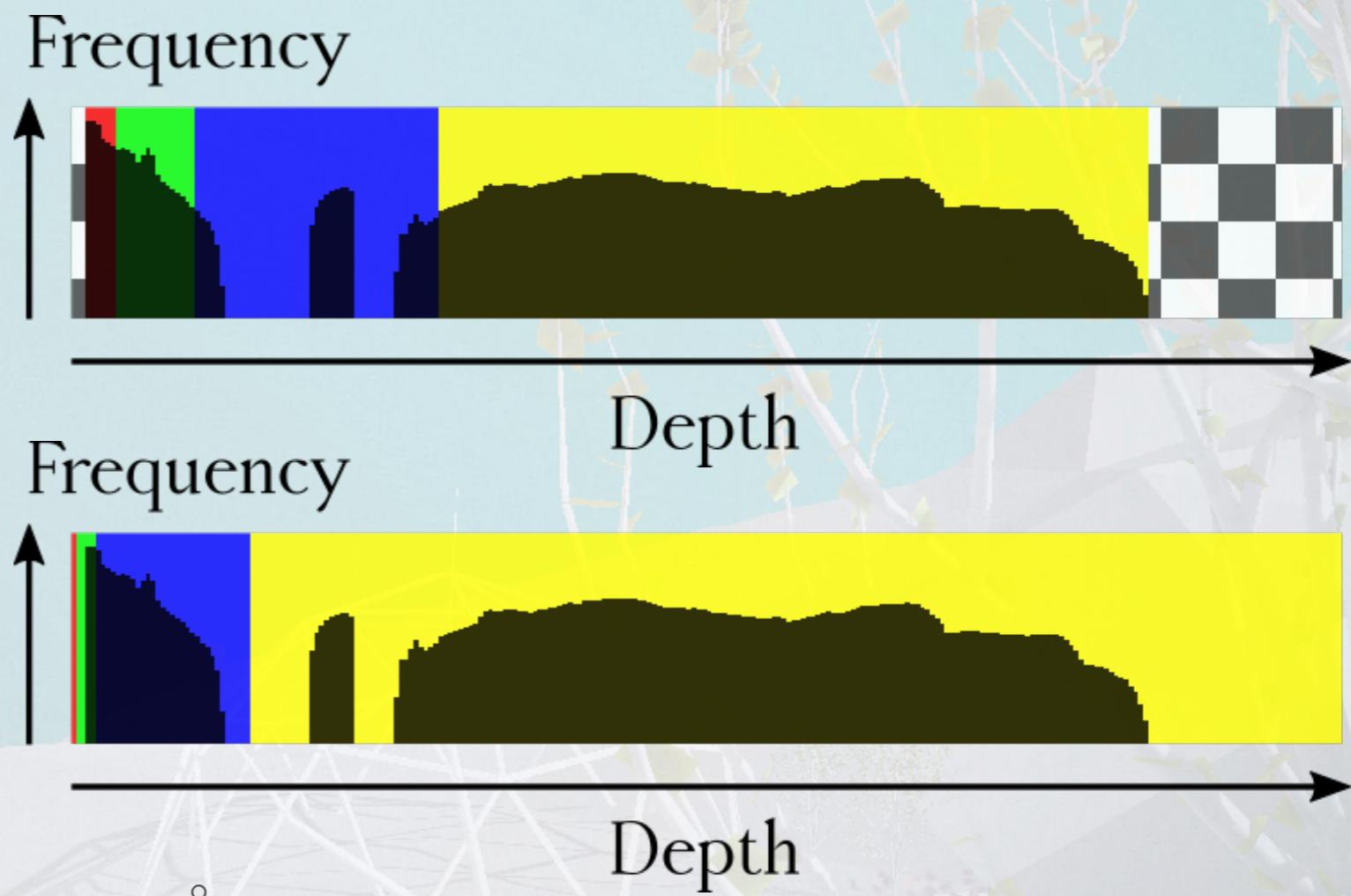
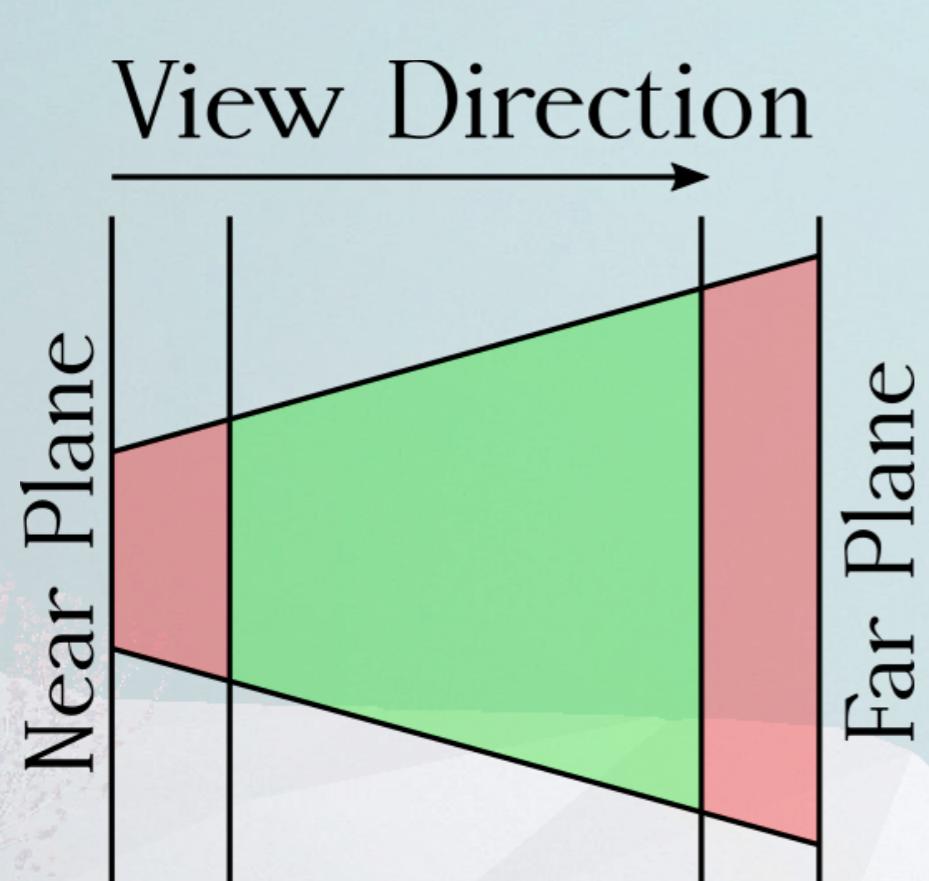
2.4 SAMPLE DISTRIBUTION SHADOW MAPS

- Betrachtet Screen Space Samples im Light Space
- Adaptive Partitionierung
- Shadow Map Frustum eng um Geometrie



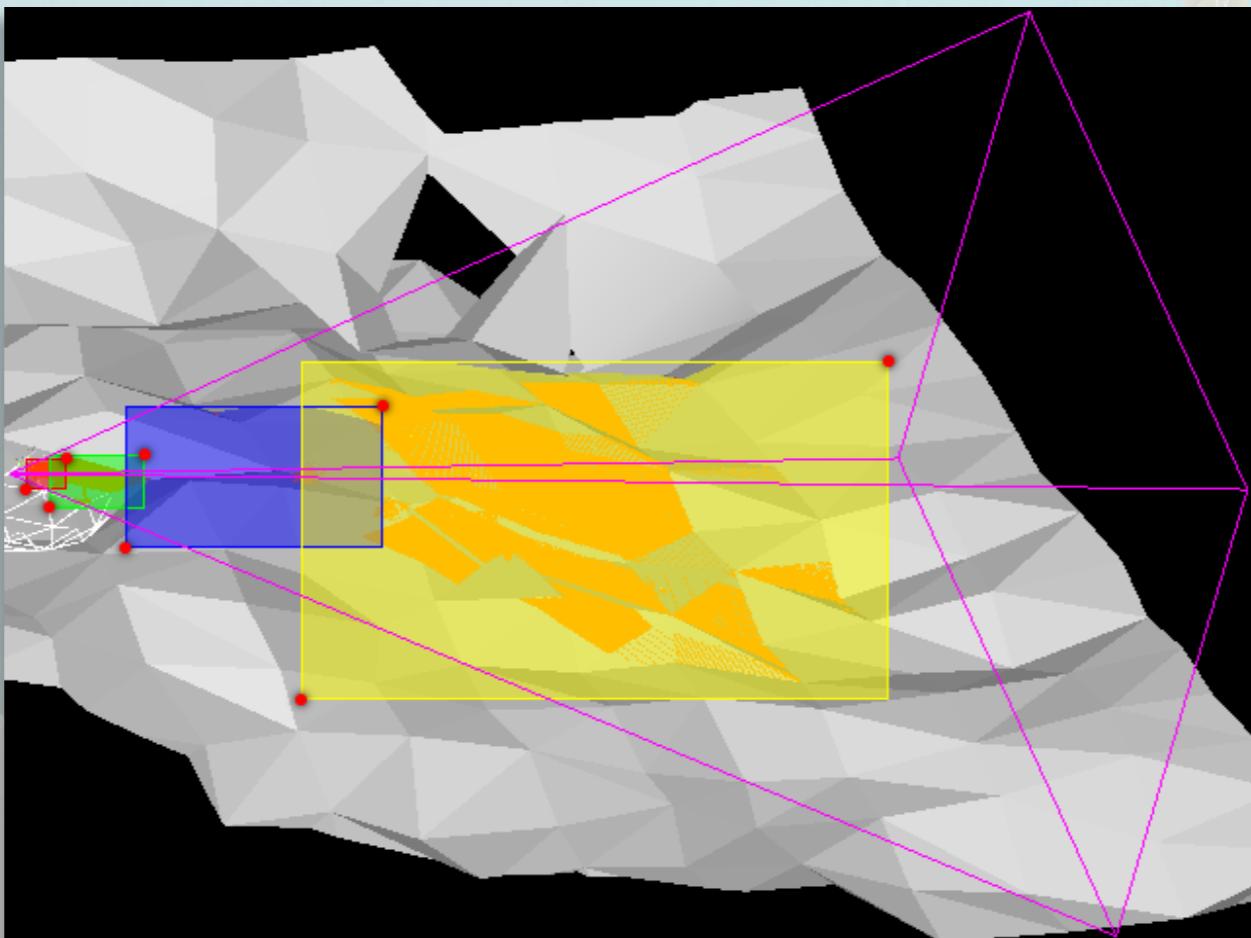
2.5 SDSM: NEAR & FAR PLANE

- Verschiebe Near- und Far-Plane dicht an die Geometrie
- Min/Max Reduktion auf Depth-Buffer

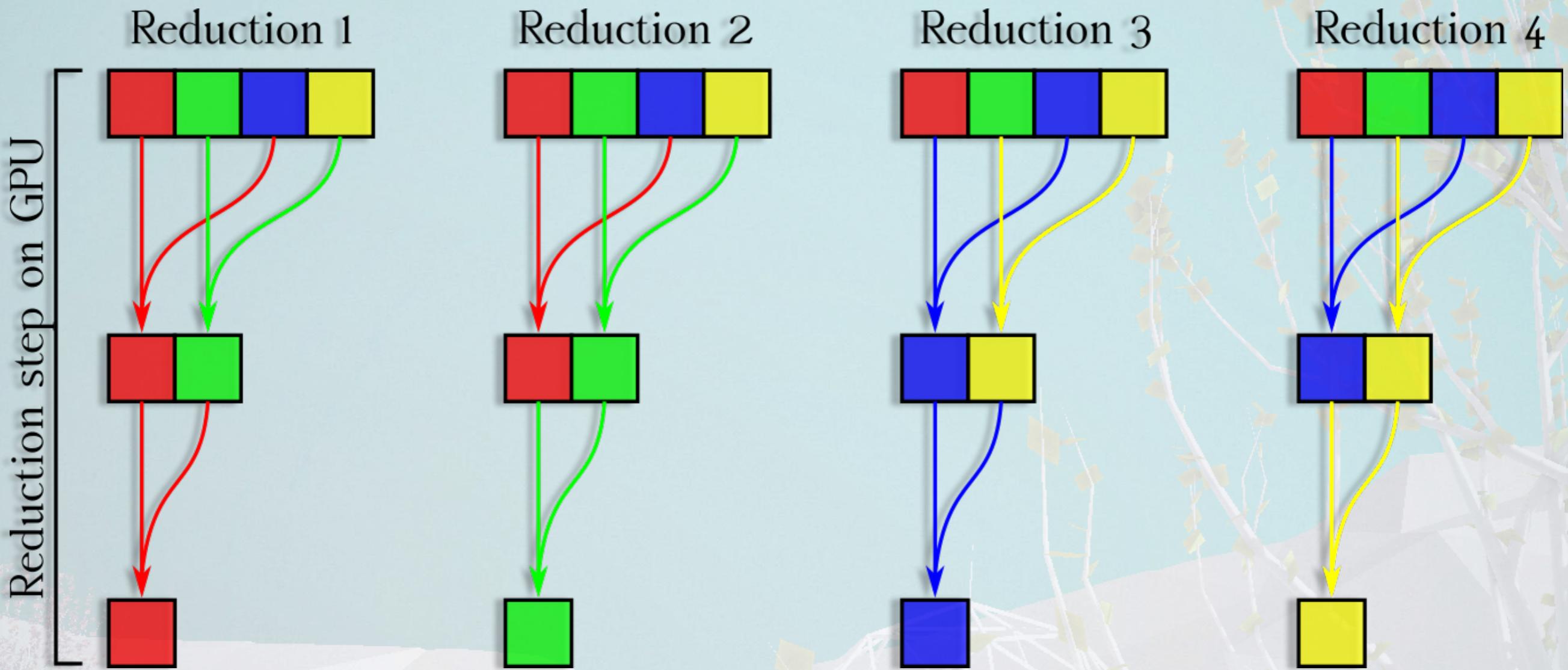


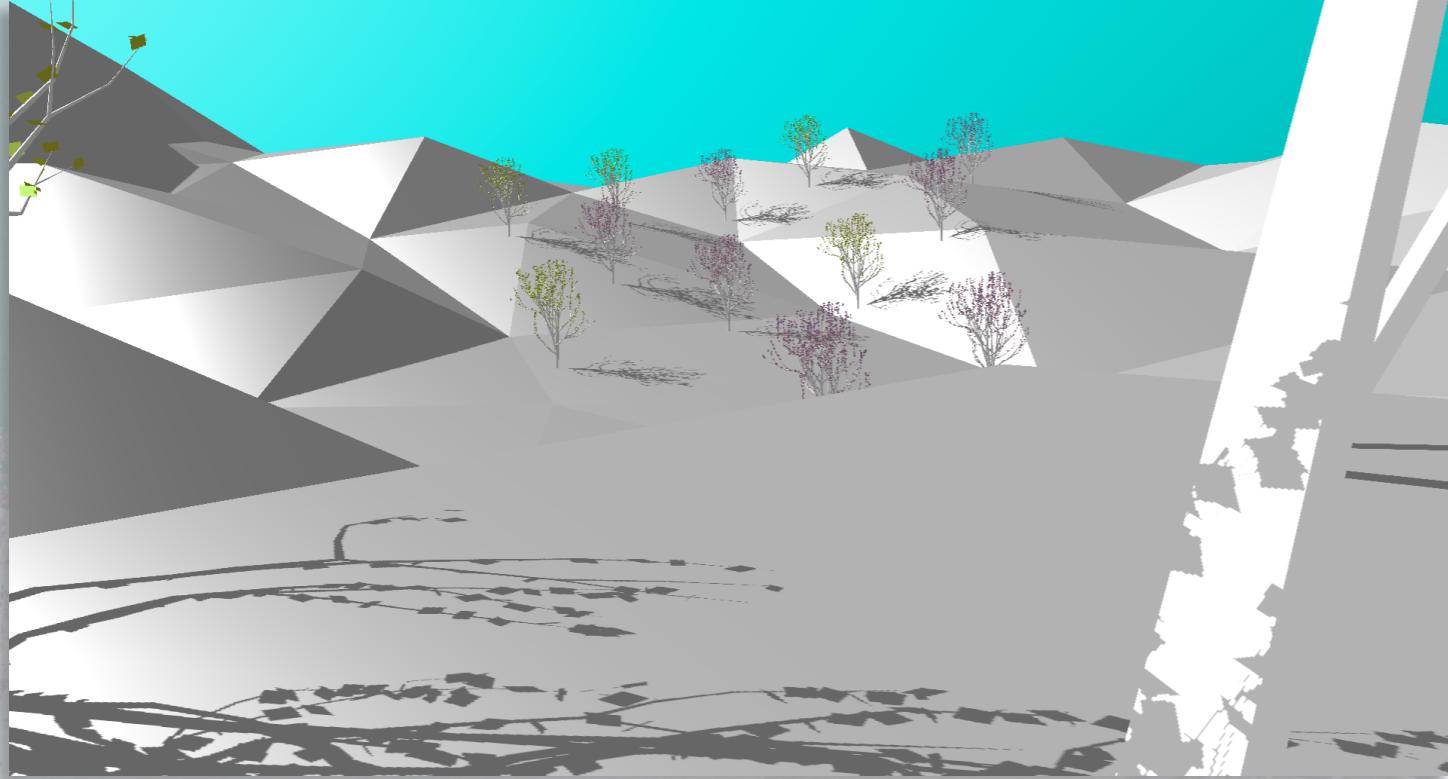
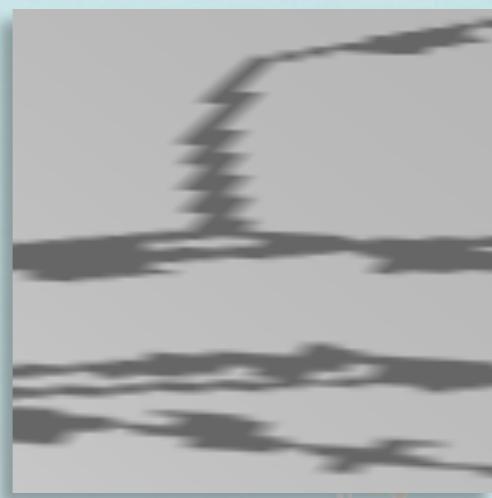
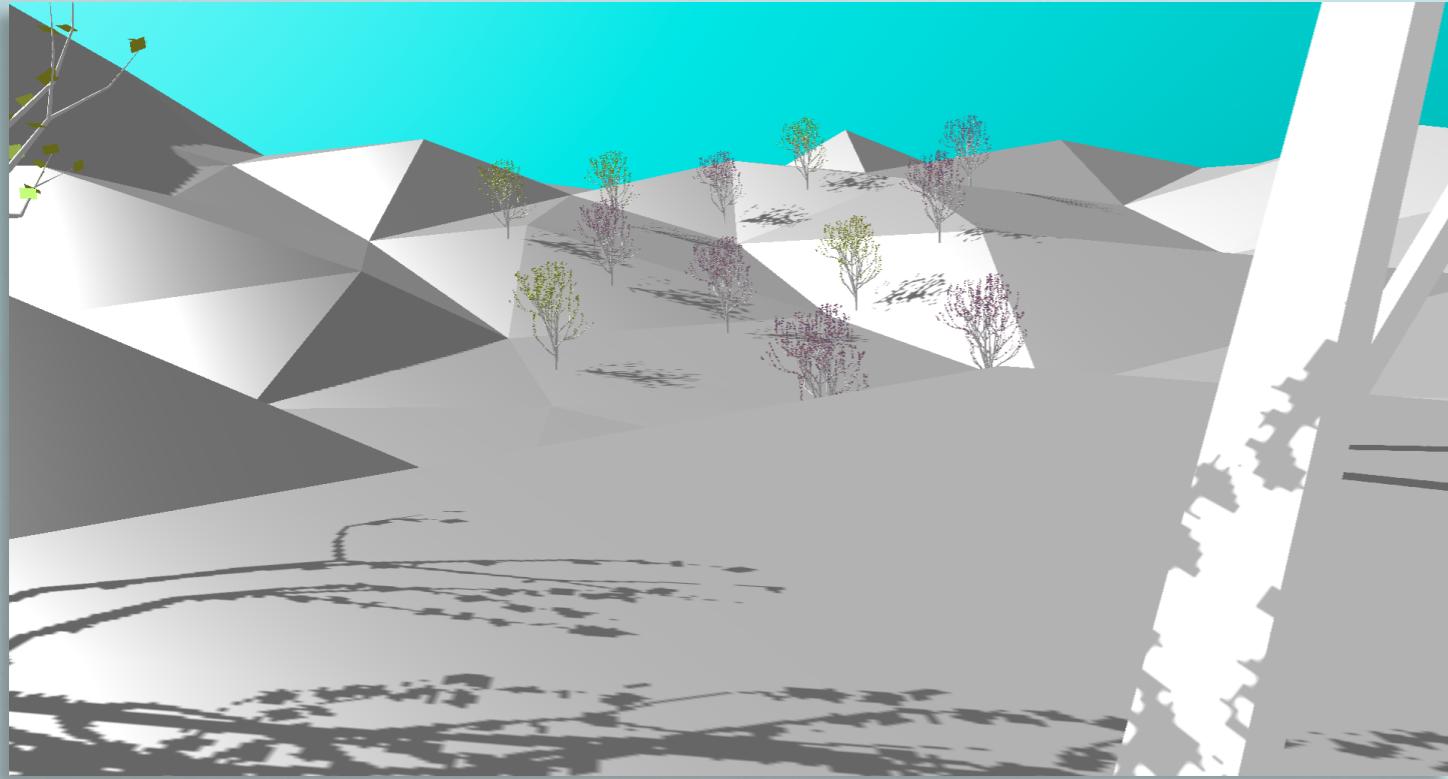
2.6 SDSM:TIGHT PARTITION FRUSTA

- Shadow Map Frusta eng um sichtbare Geometrie
- Berechne AABB der sichtbaren Geometrie per Reduktion

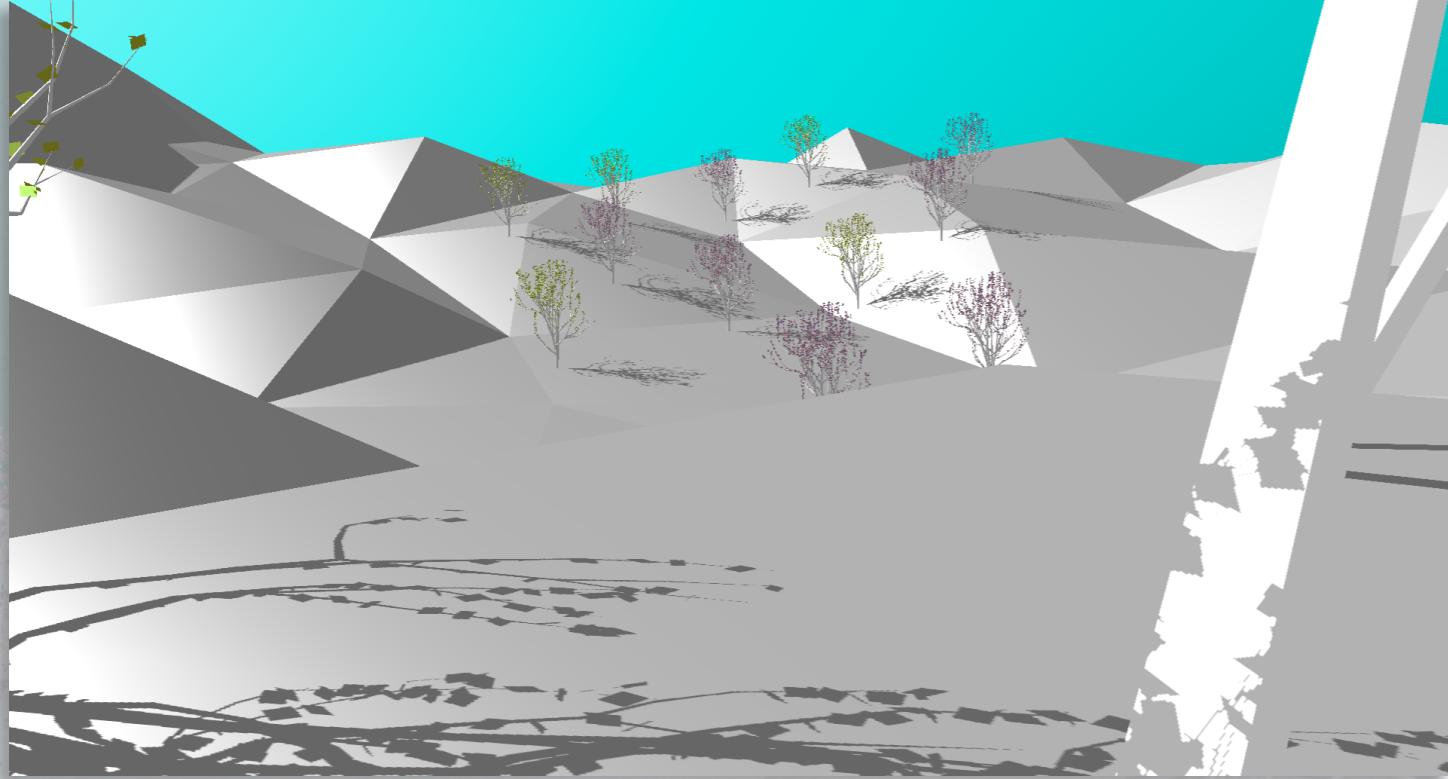
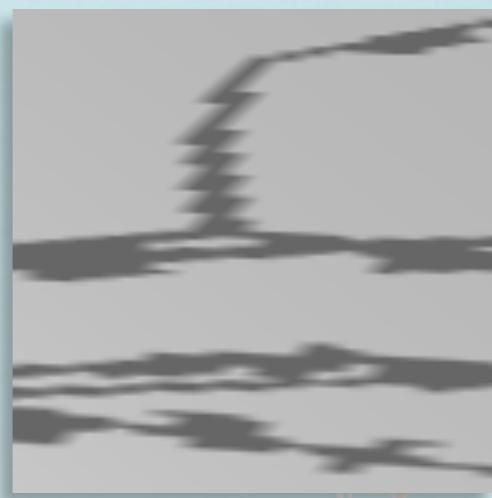
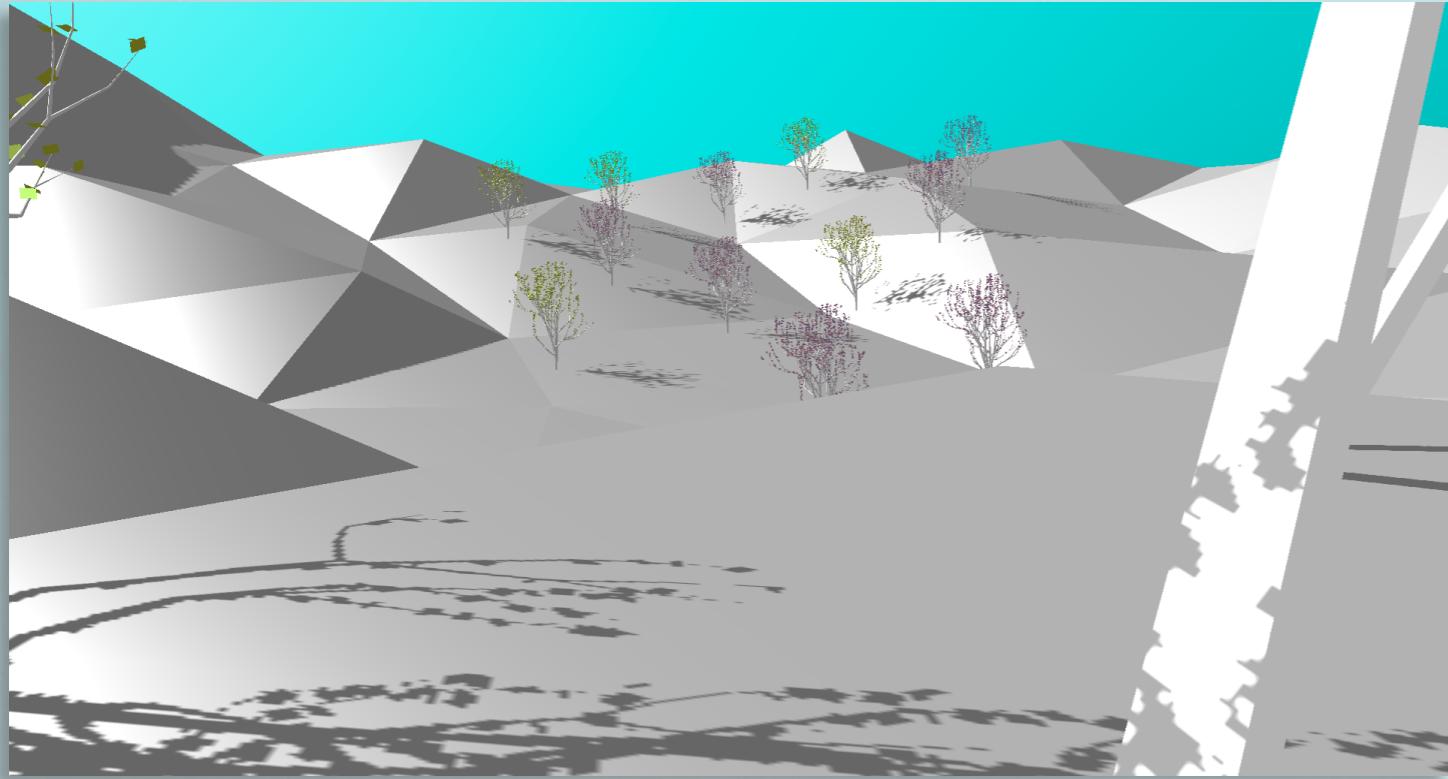


2.7 SDSM: REDUCTION



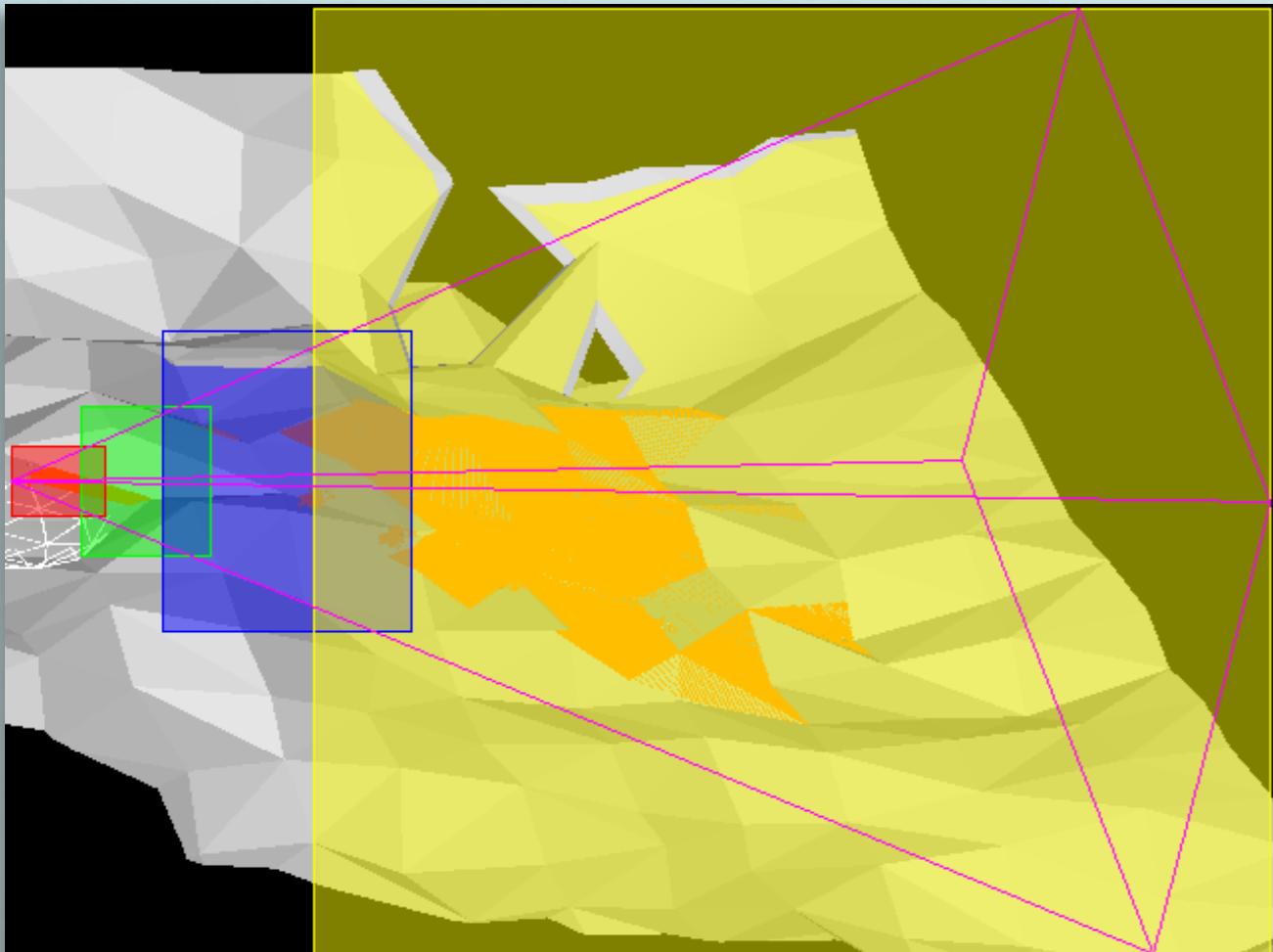


PSSM

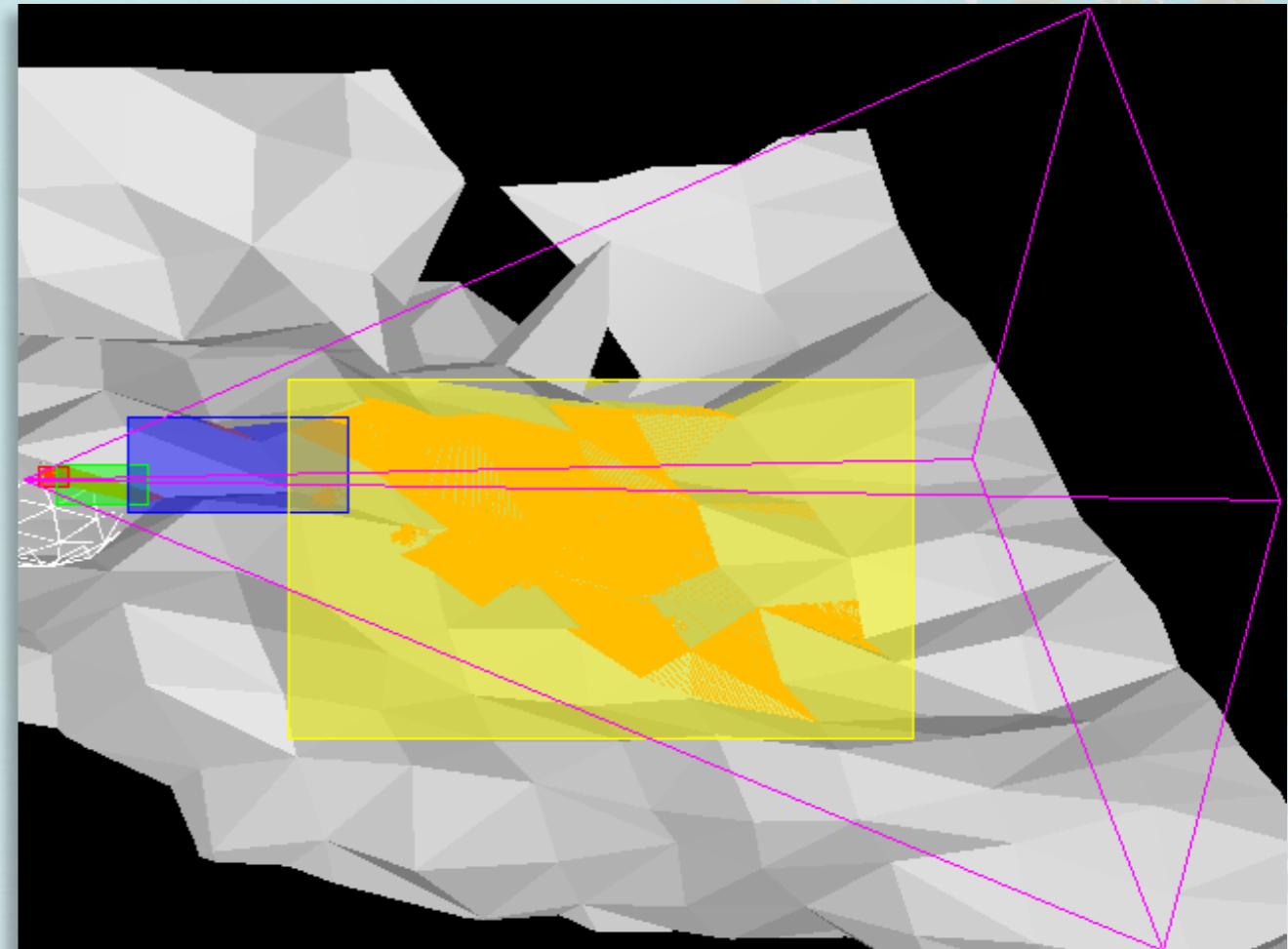


SDSM

2.8 SDSM: LIGHT SPACE FRUSTA

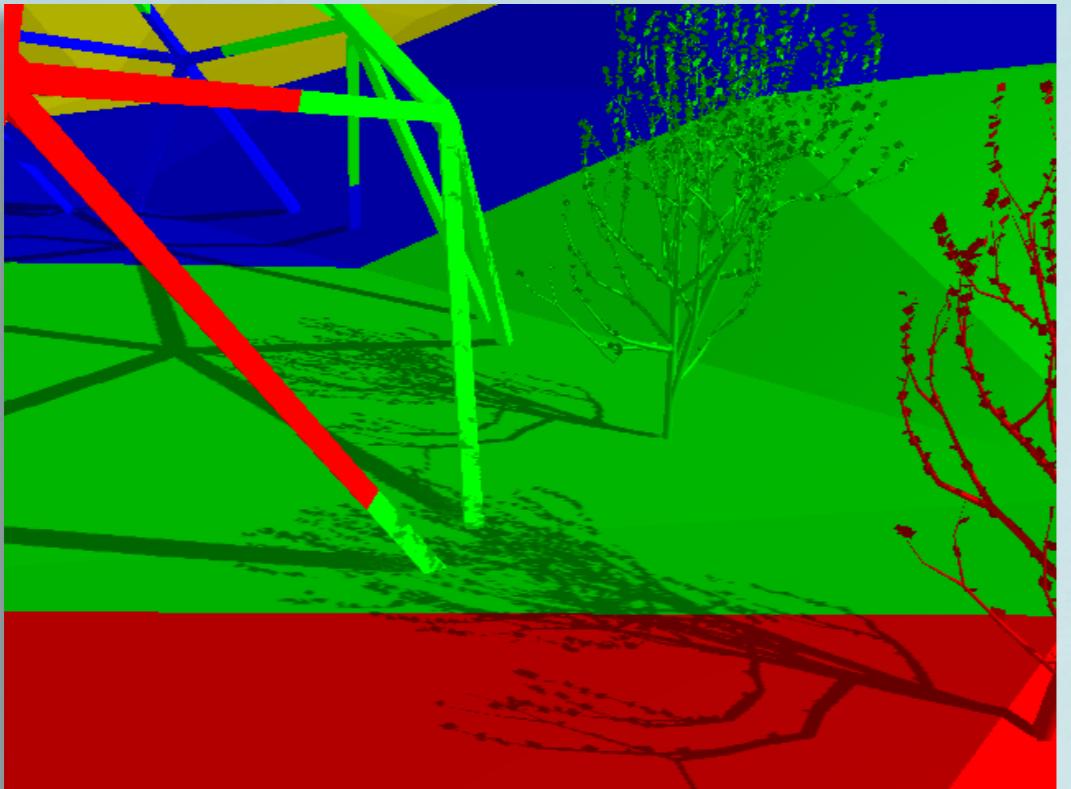


PSSM

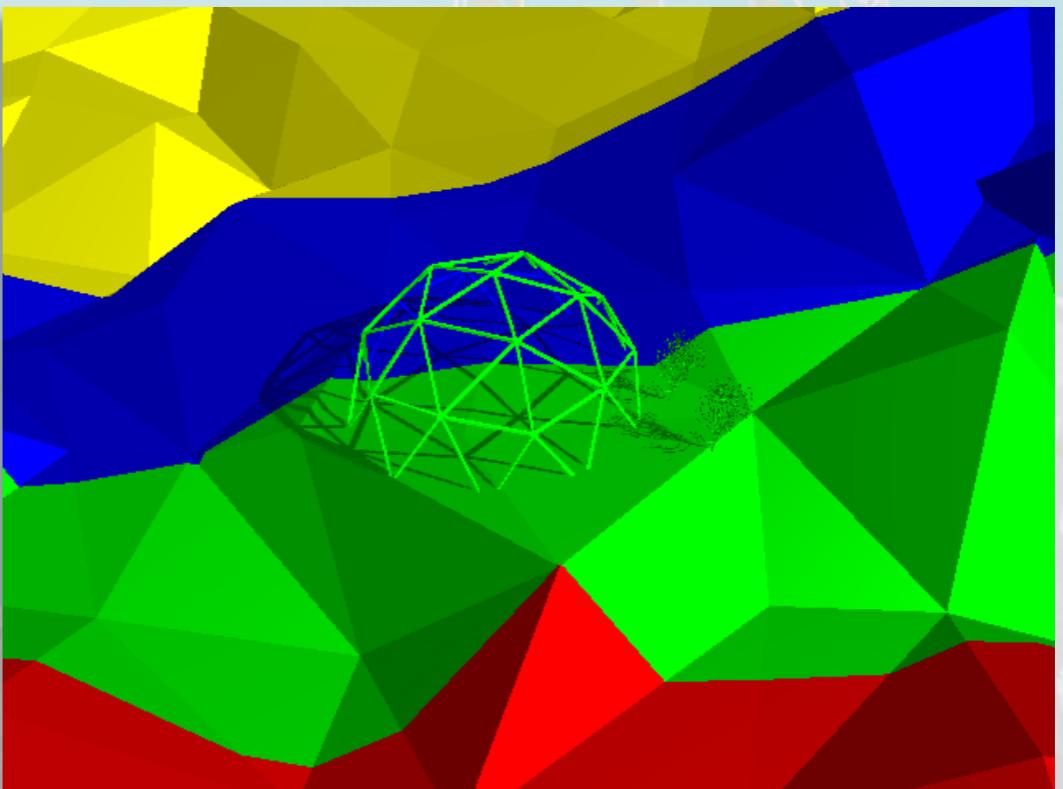
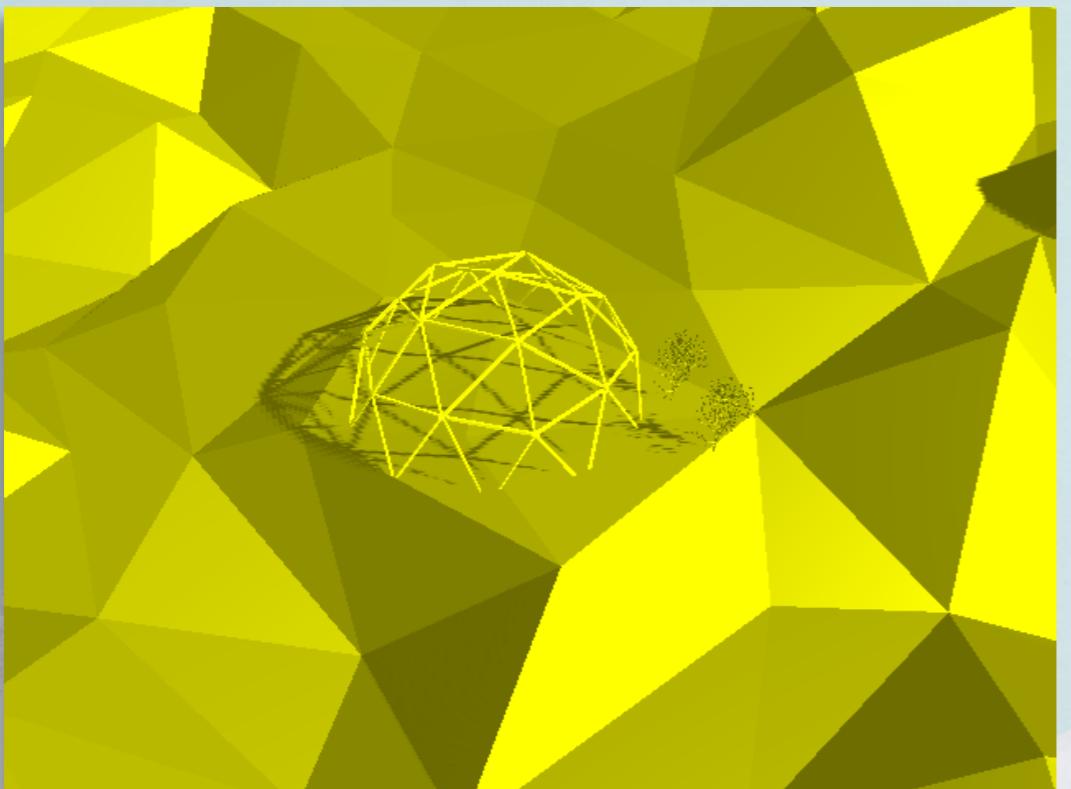
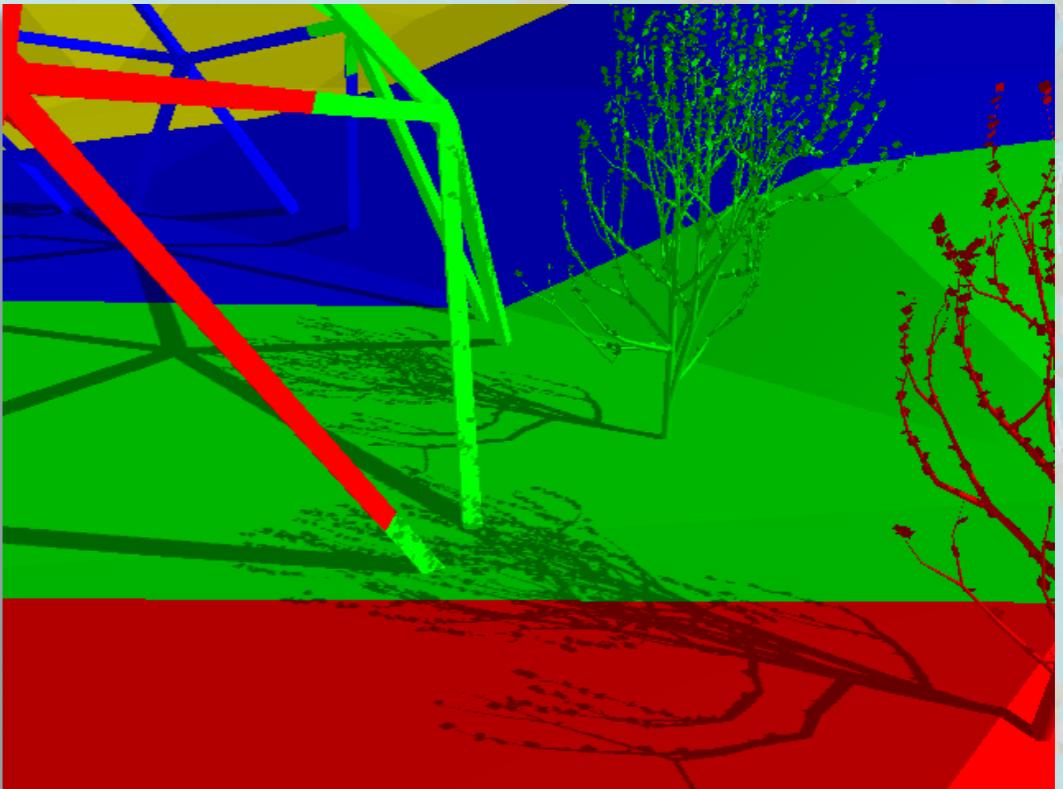


SDSM

PSSM

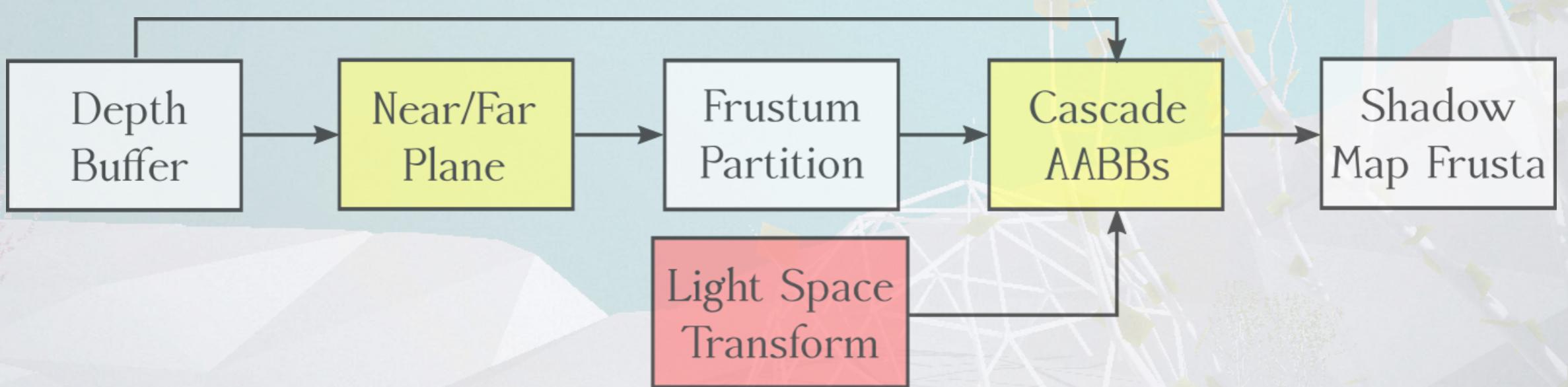


SDSM

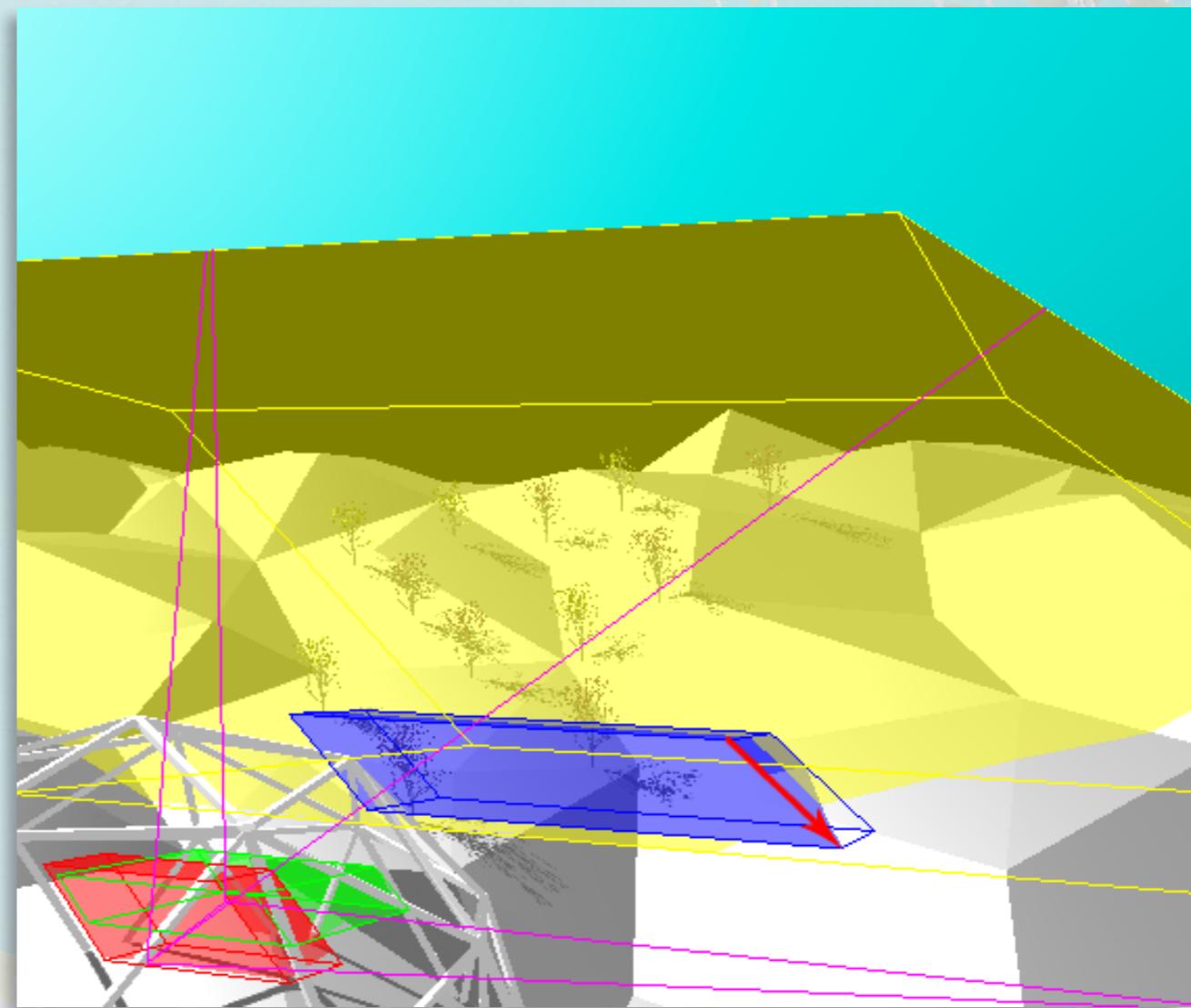
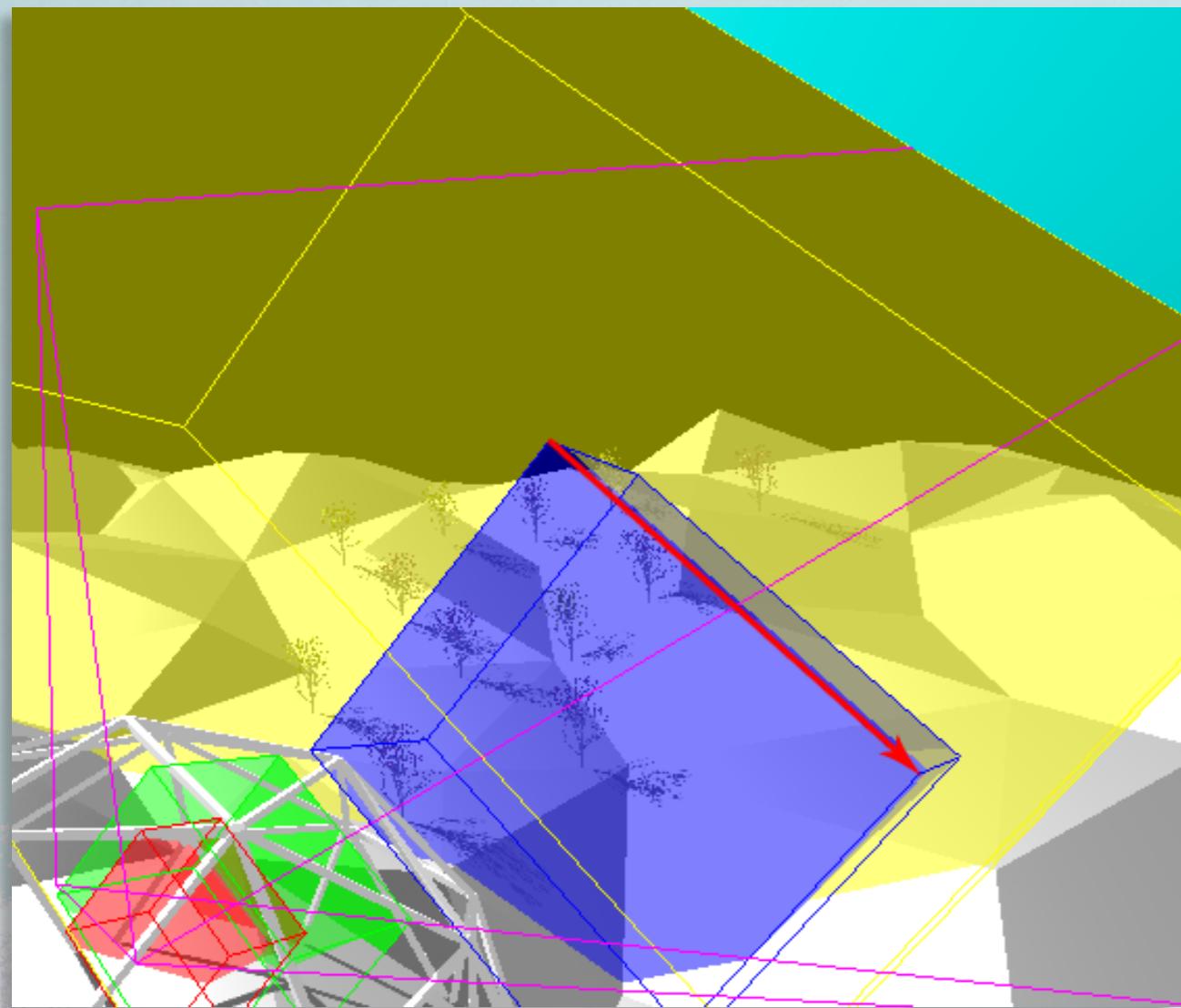


2.9 SDSM: ERGEBNISSE

- SDSM ist sehr robust
- Zuverlässige Ergebnisse
- Screenspace Algorithmus
- Vollständig auf GPU implementierbar

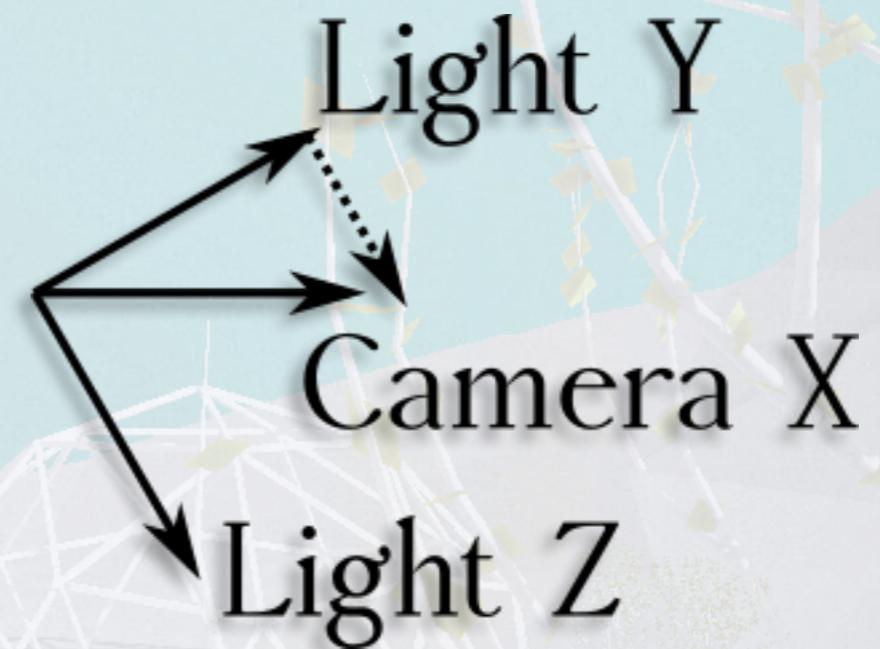
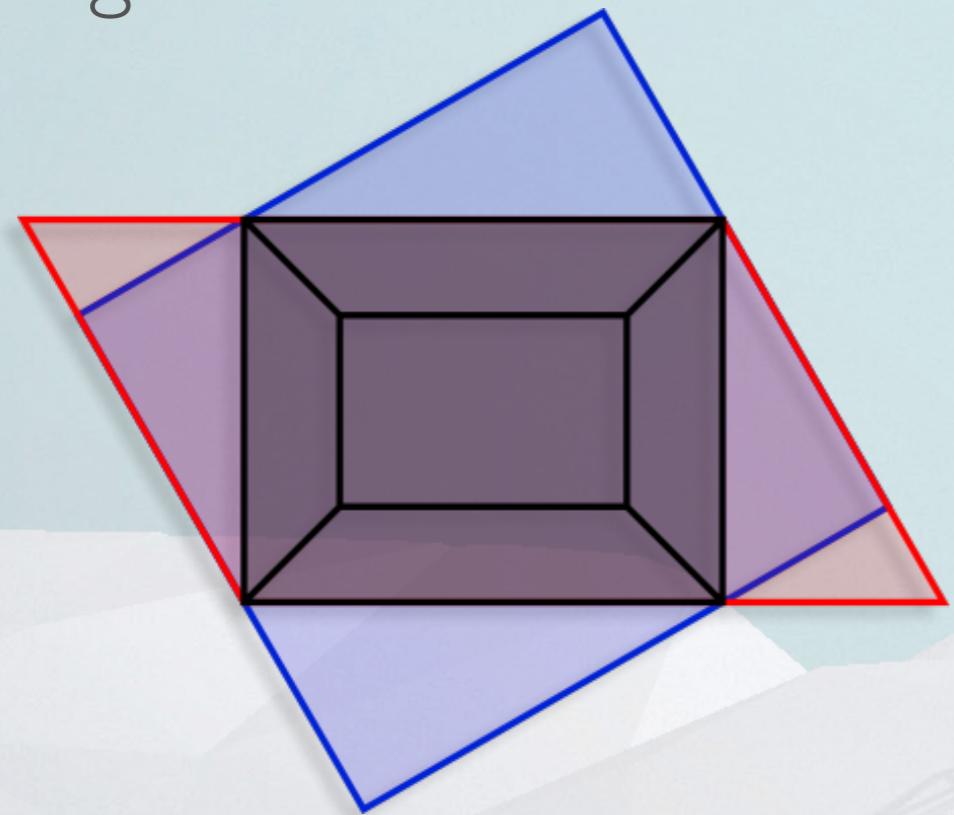


2.10 SDSM: SHEARED SAMPLE DISTRIBUTION SHADOW MAPS



2.10 SDSM: SHEARED SAMPLE DISTRIBUTION SHADOW MAPS (2)

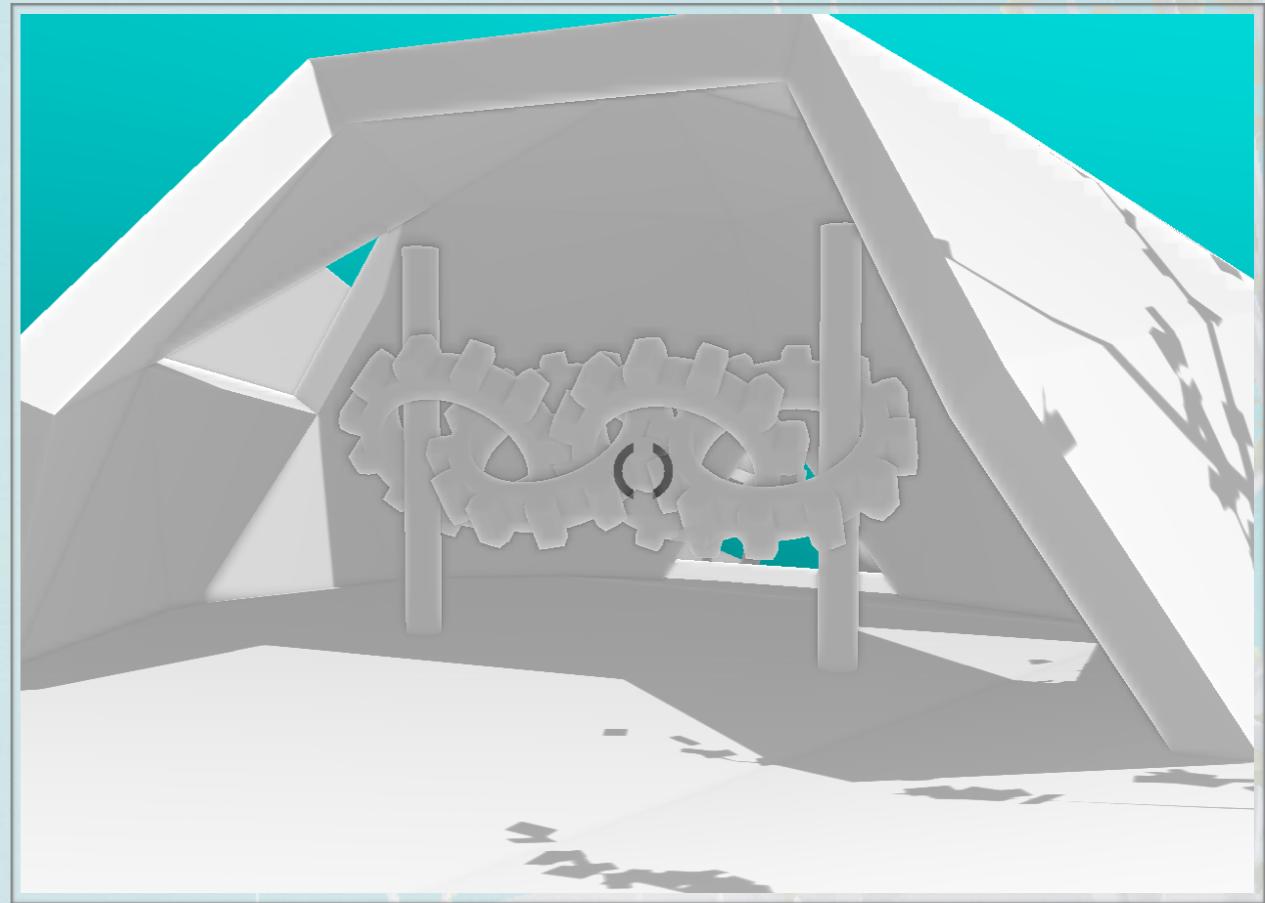
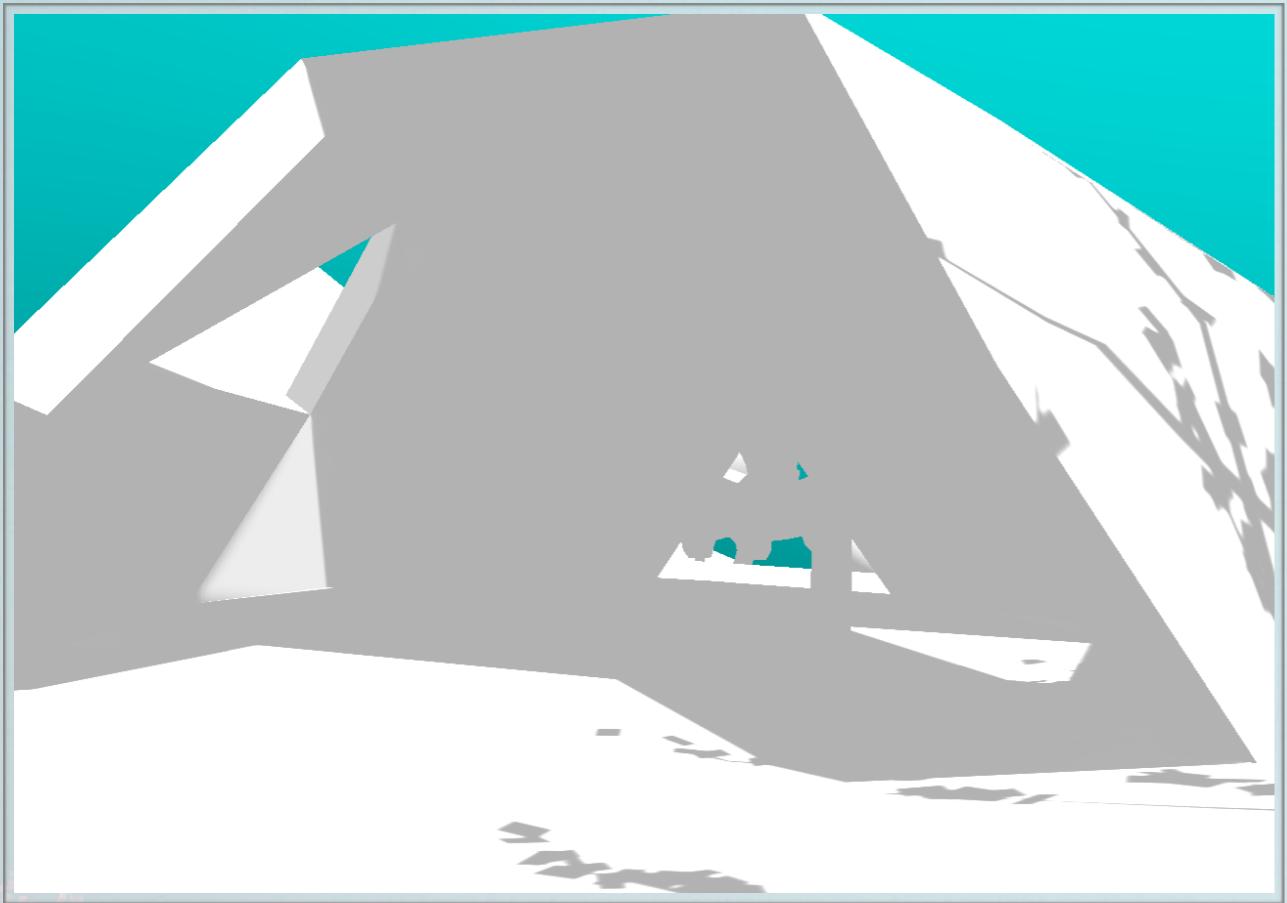
- Szene oft sehr flach, geneigtes Licht
- Horizontale Near- und Farplane der Shadow Map
- Unausgereift



3. VOLUMETRIC OBSCURANCE

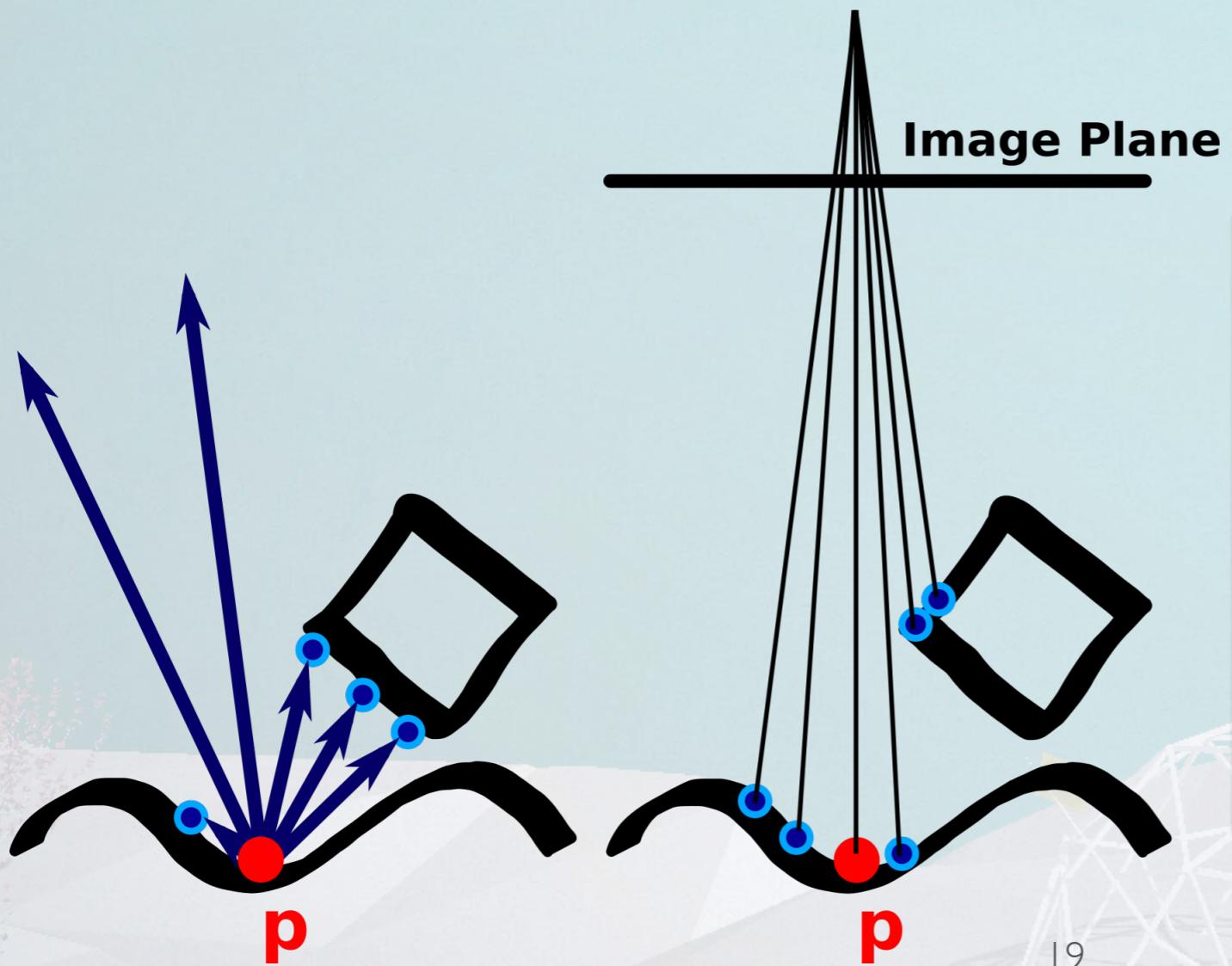
- Global Illumination Technik
- „Wo viel Geometrie ist, ist auch viel Schatten“
- Ambient Occlusion basiert auf diesem Prinzip
- Volumetric Obscurrence bietet Alternative zu AO

3. I VO: MOTIVATION



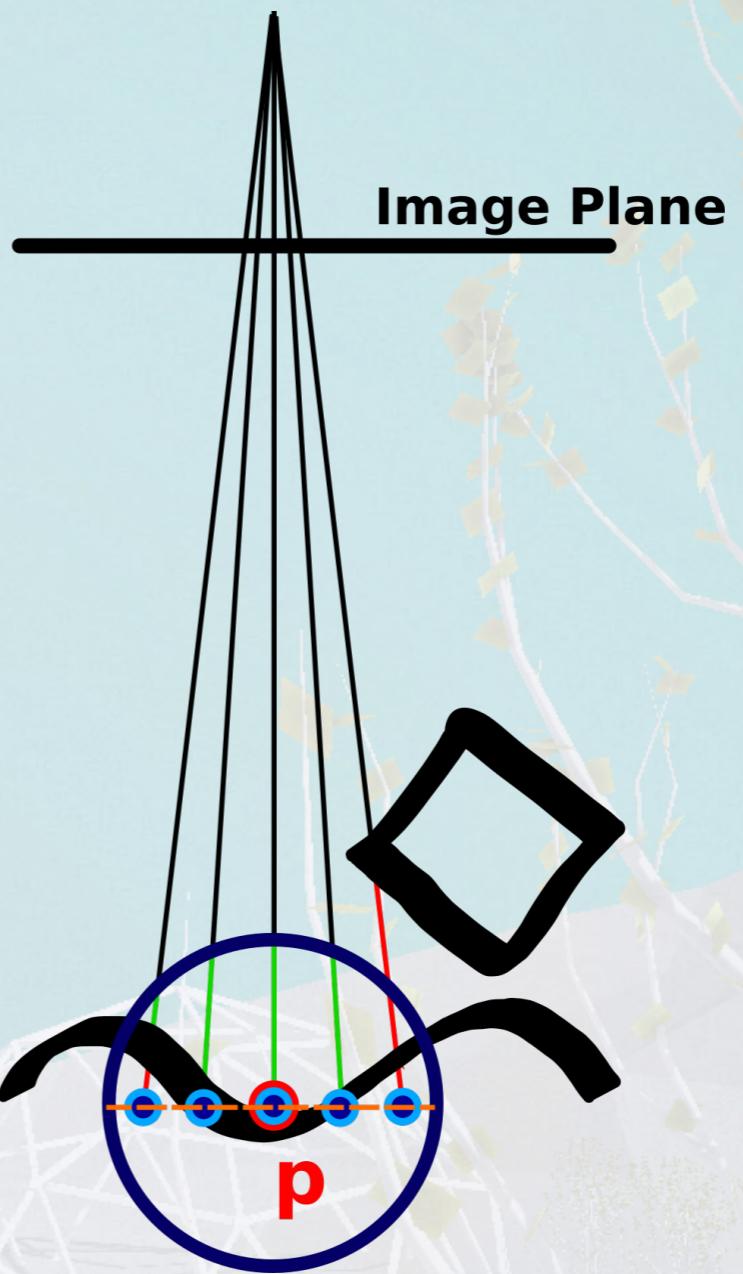
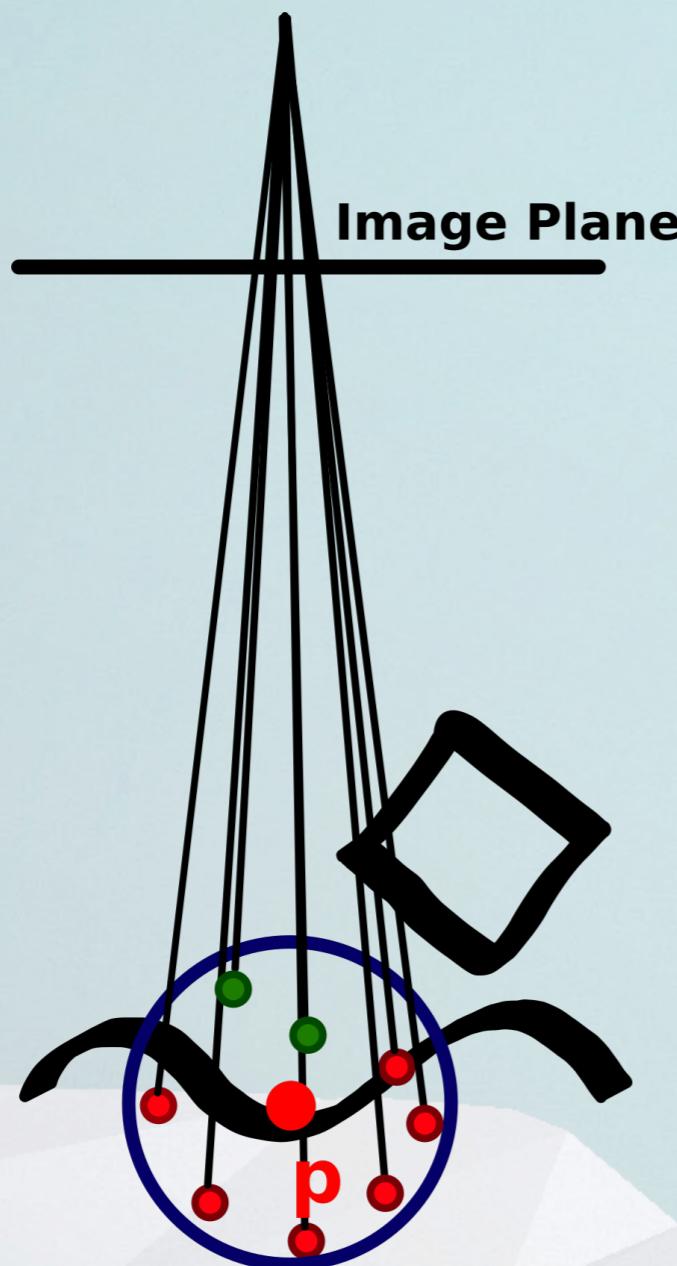
3.2 VO: AMBIENT OCCLUSION

- Ambient Occlusion Ansätze:
 - (Global) Ambient Occlusion & Screenspace Ambient Occlusion



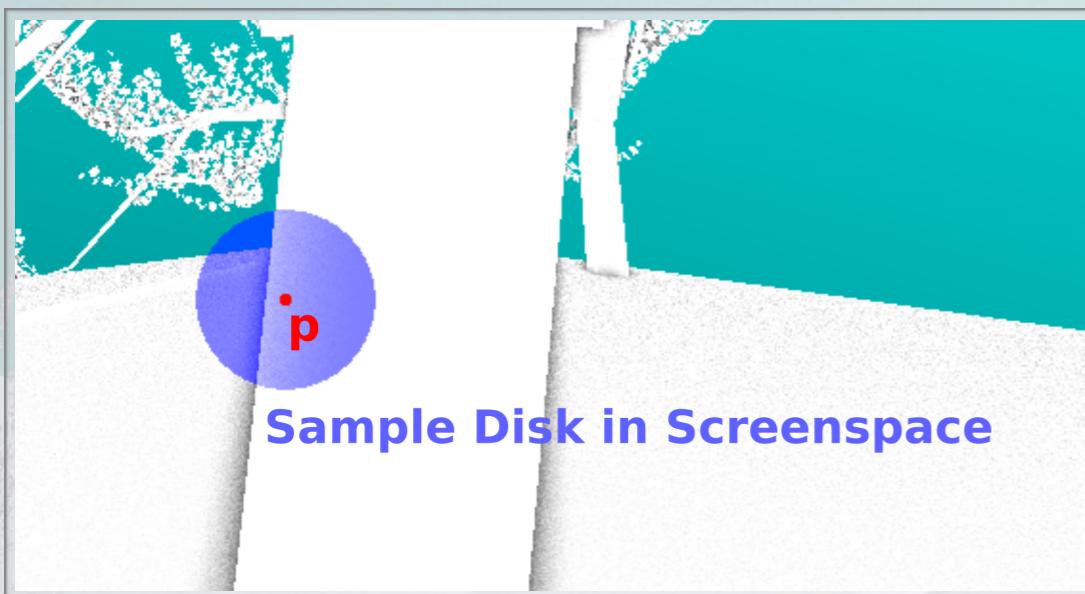
3.3 VOLUMETRIC OBSCURANCE

- Point Samples und Line Samples



3. 4 VO: LINE SAMPLING

- Betrachte für jeden Texel den korrespondierenden Punkt
- Sample Tiefenbuffer in Screenspace projizierter Umgebung
- Berechne verdecktes Volumen



3.5 VO: LINE SAMPLING ALGORITHMUS

Line Sampling Pseudo Code

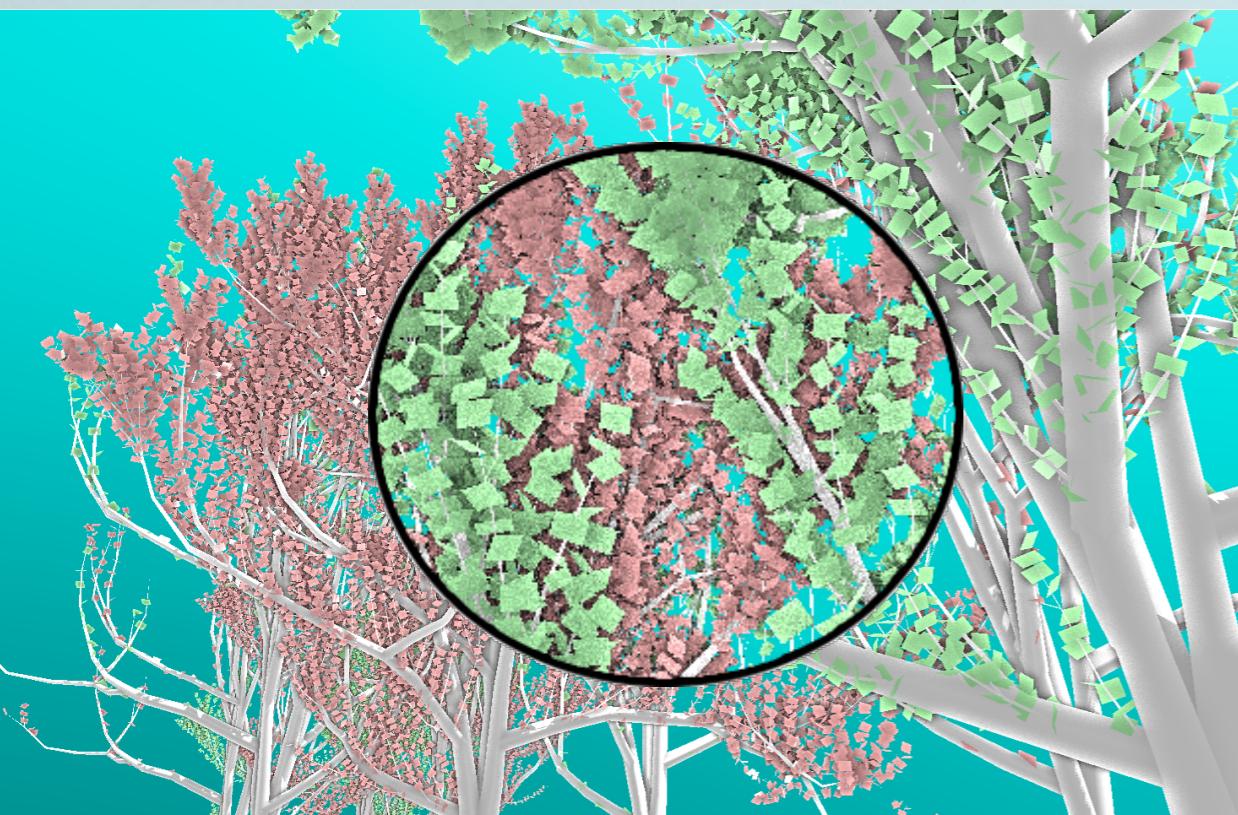
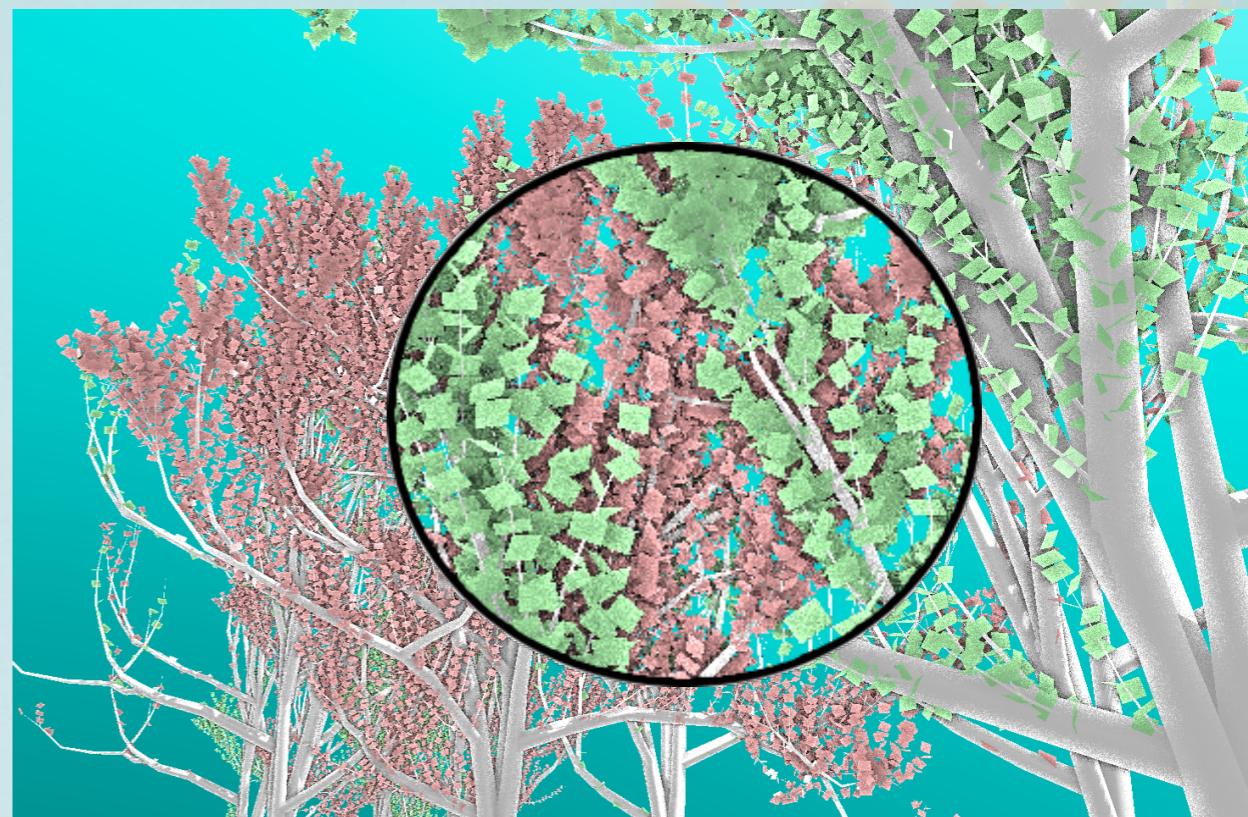
Function *LineSampling(nSamples, depth)*

```
    sumSamples = 0;  
    sumVolume = 0;  
    for i = 0 → nSamples - 1 do  
        radius =  $\sqrt{\text{rand}()}$ ;  
         $\alpha$  =  $\text{rand}() \cdot 2\pi$ ;  
         $v_i = \sqrt{1 - \text{radius}^2}$ ;  
         $p = (\cos(\alpha) \cdot \text{radius}, \sin(\alpha) \cdot \text{radius})^T$ ;  
        sumSamples = sumSamples +  $v_i \cdot \text{SamplePoint}(p, \text{depth})$ ;  
        sumVolume = sumVolume +  $v_i$ ;  
    end  
    return  $\frac{\text{sumSamples}}{\text{sumVolume}}$  ;  
end
```

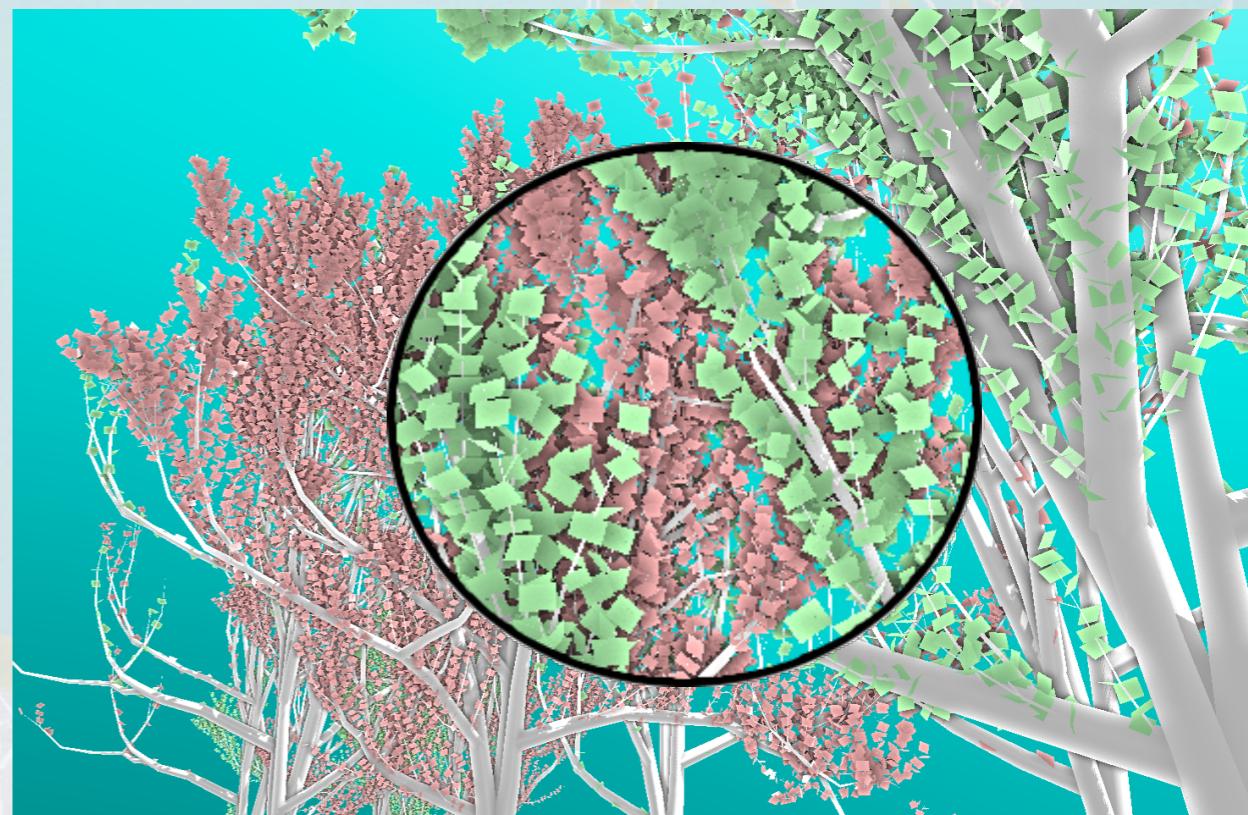
1 Samples



10 Samples



100 Samples

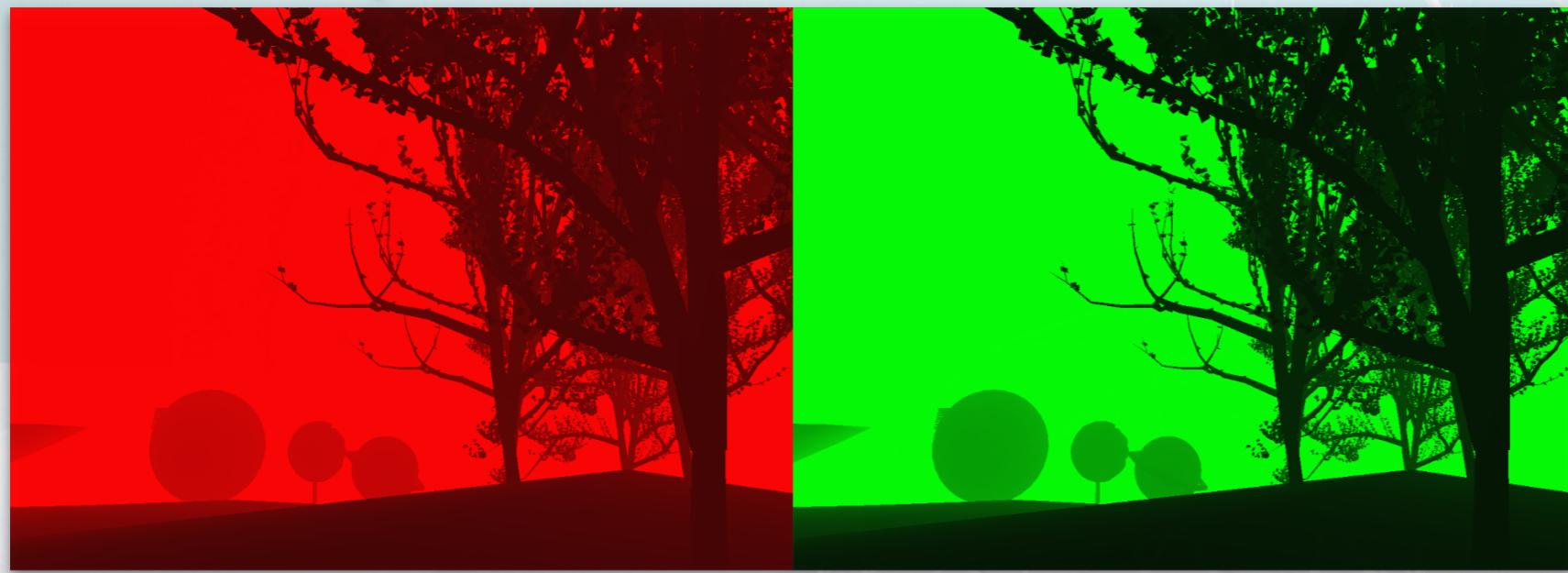


1000 Samples

3.6 VO: AREA SAMPLING

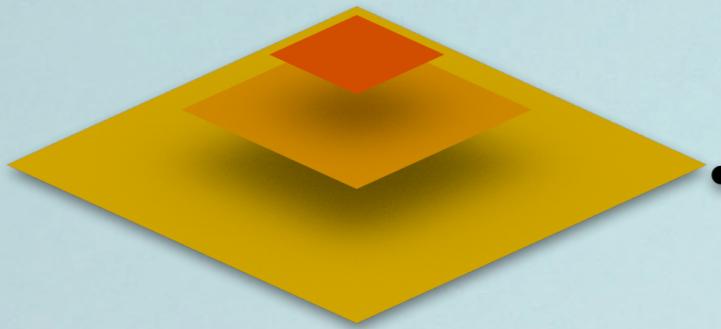
- Line Sampling teuer, da viele Samples pro Pixel
- Statistische Beschreibung die eine Approximation der uniformen Tiefenverteilung ermöglicht
- VSM bietet gleiches Prinzip und kann auf VO angewendet werden

Depth (z)

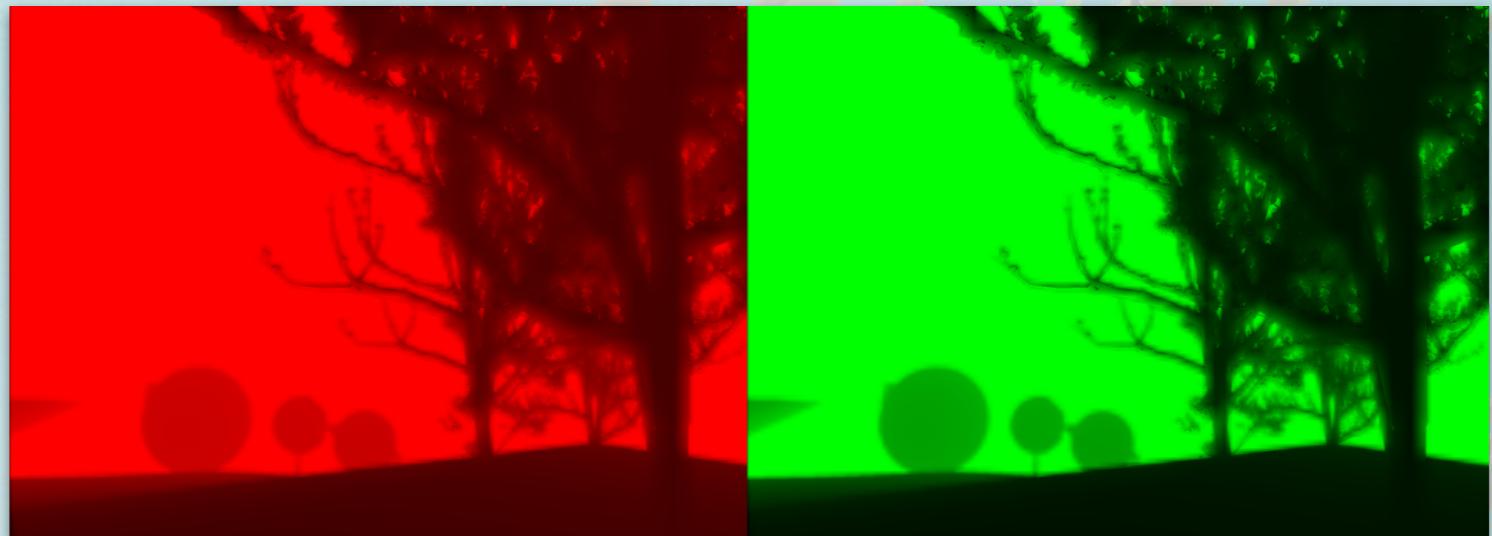


generate Mipmaps





Mipmaps

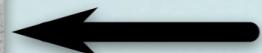


$$\mu = R \quad \sigma = \sqrt{G - \mu^2}$$



obscurance Term

$$V_c(z_0, z_1, a, b) = a \cdot (z_1^2 - z_0^2)/2 + b \cdot (z_1 - z_0)$$
$$a = \frac{-1}{2 \cdot \sigma}; \quad b = -a \cdot (\mu + \sigma)$$



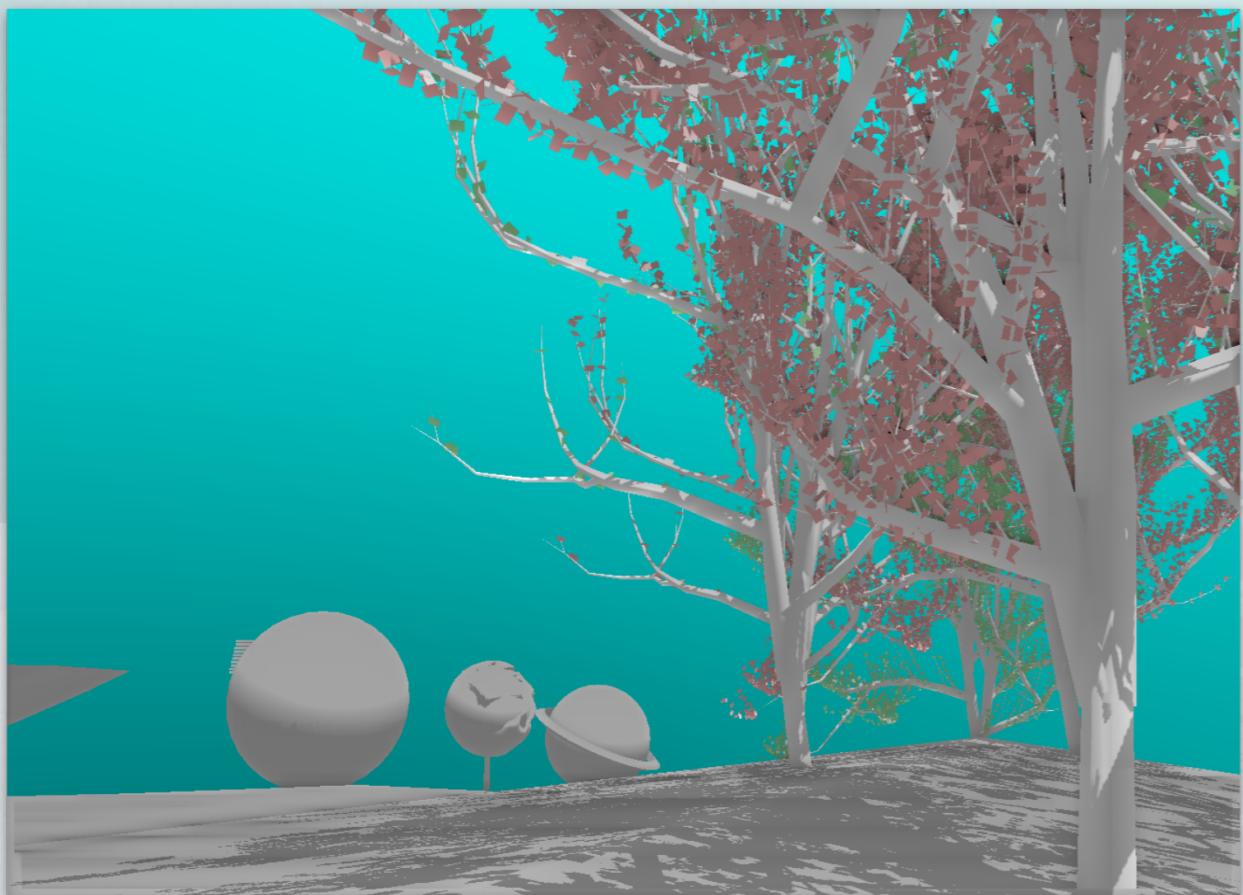
mean



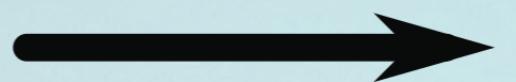
standard deviation



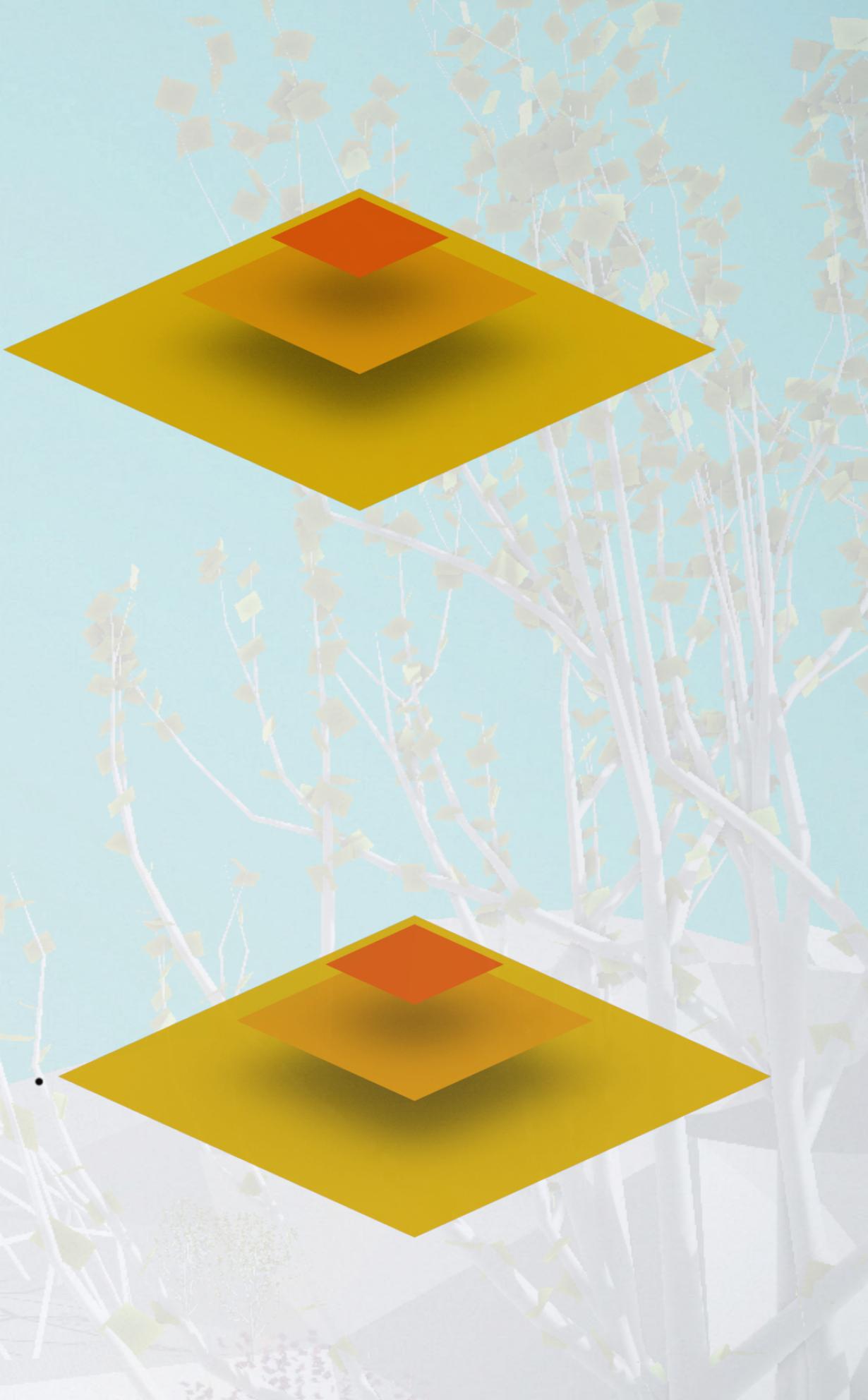
obscurance Term

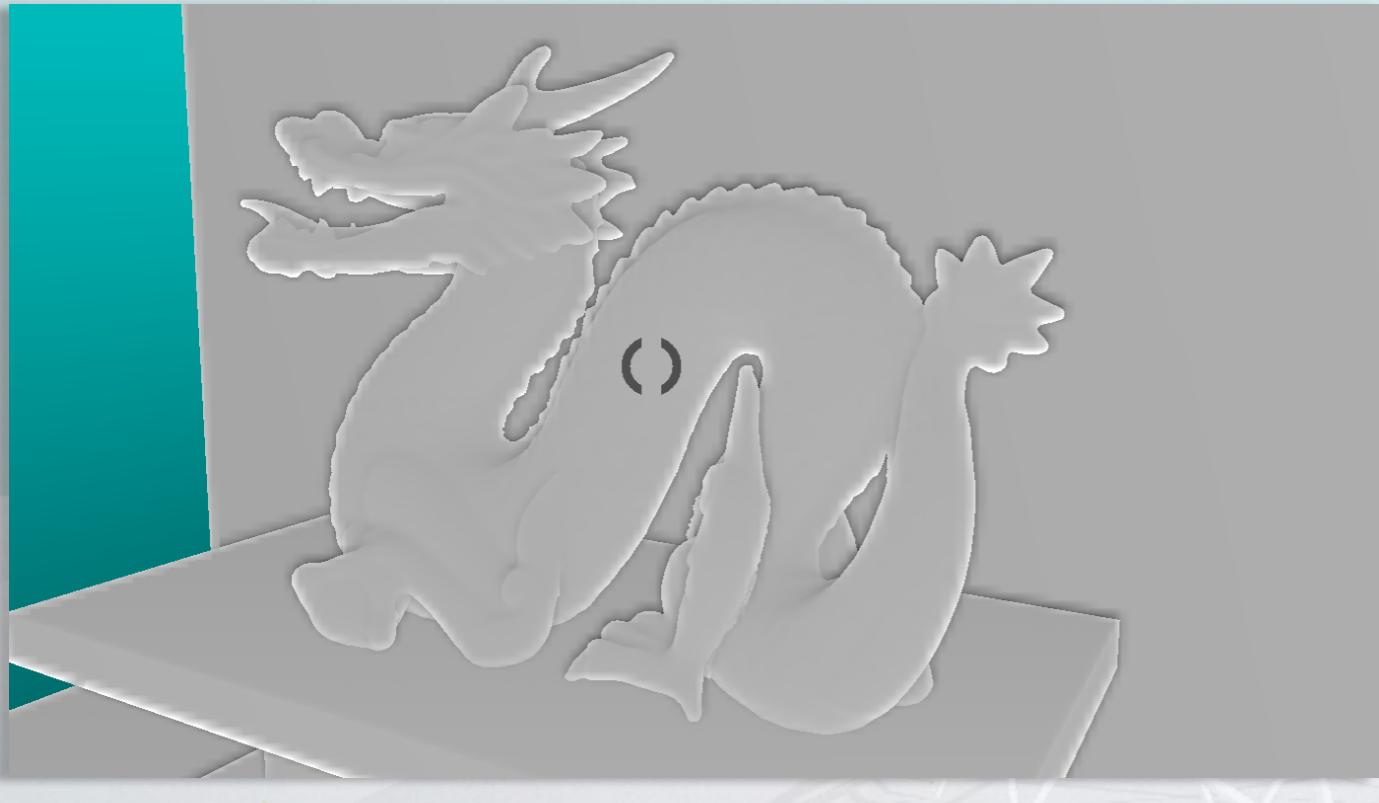
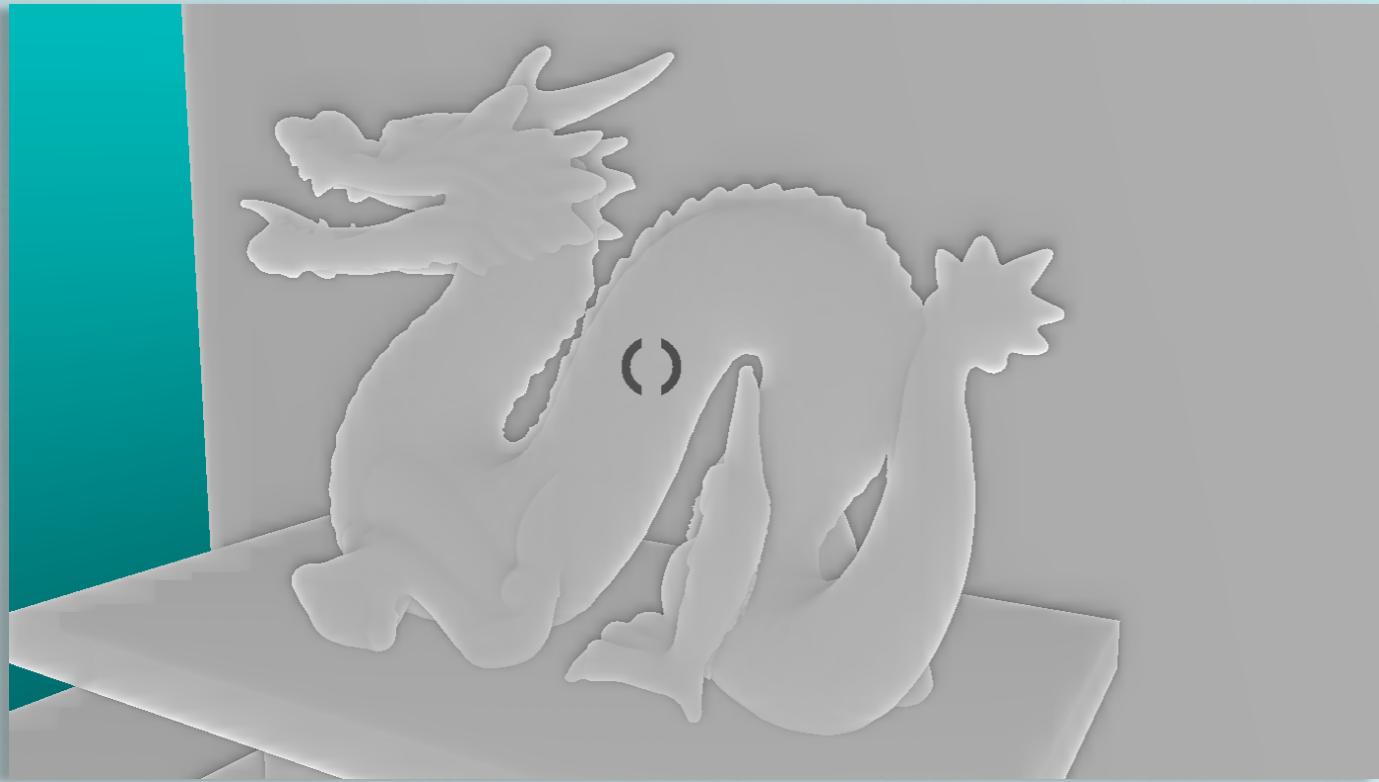


verrechnetes Bild

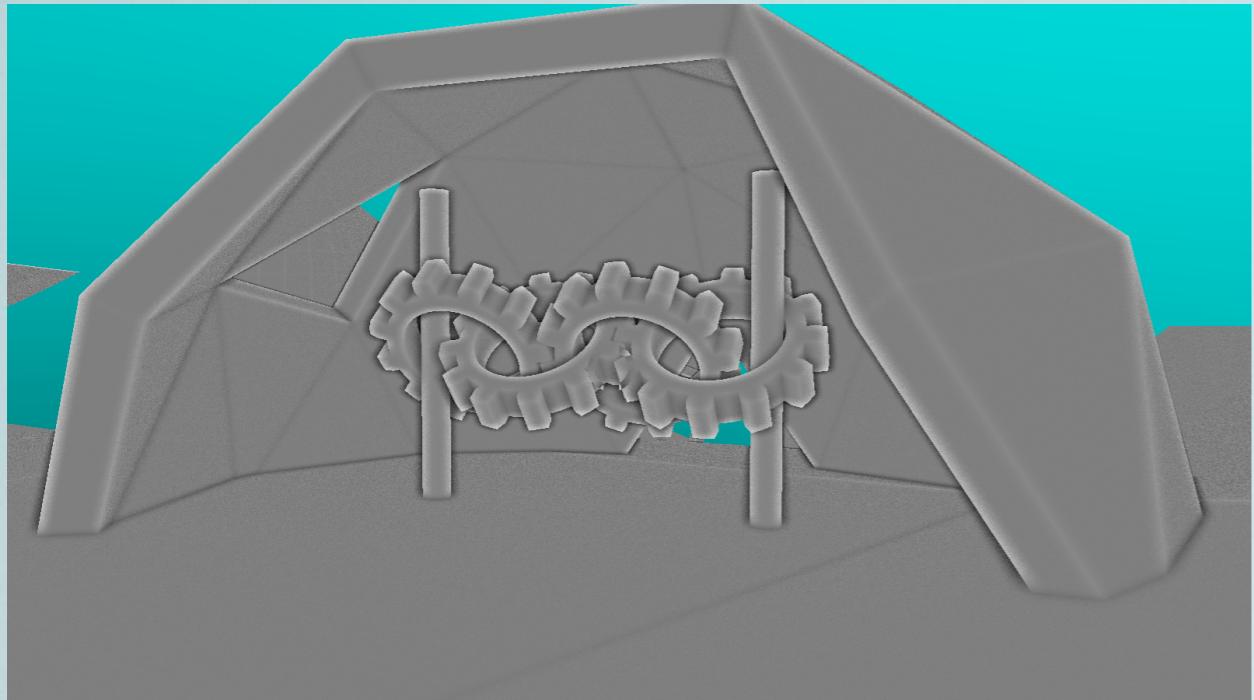


$$= \frac{1}{16} \cdot \begin{pmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{pmatrix} \cdot$$

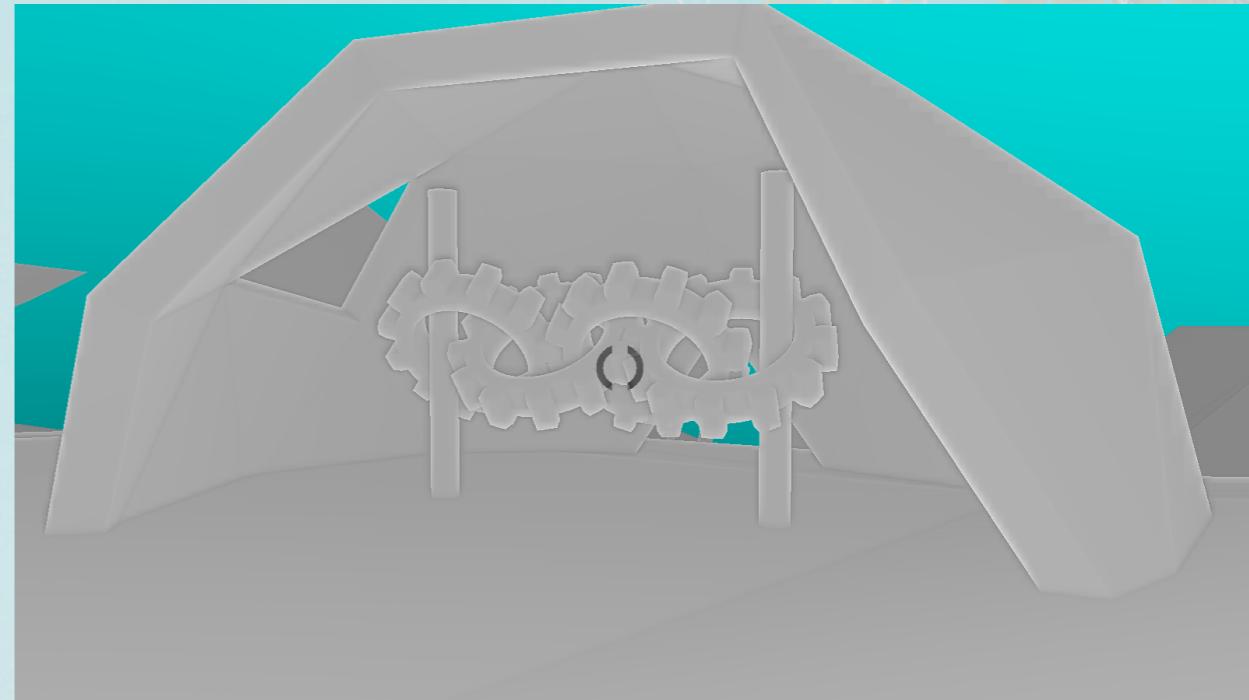




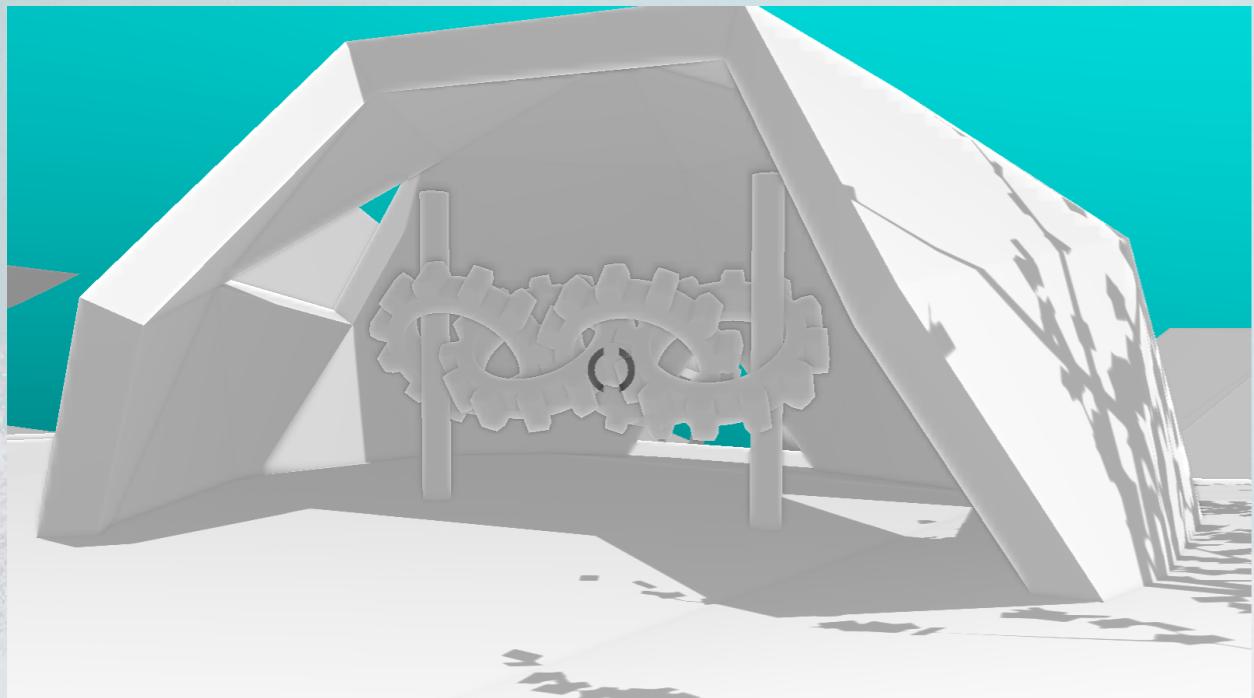
3.7 VO: ERGEBNISSE



Line Sampling 100 Samples



Area Sampling



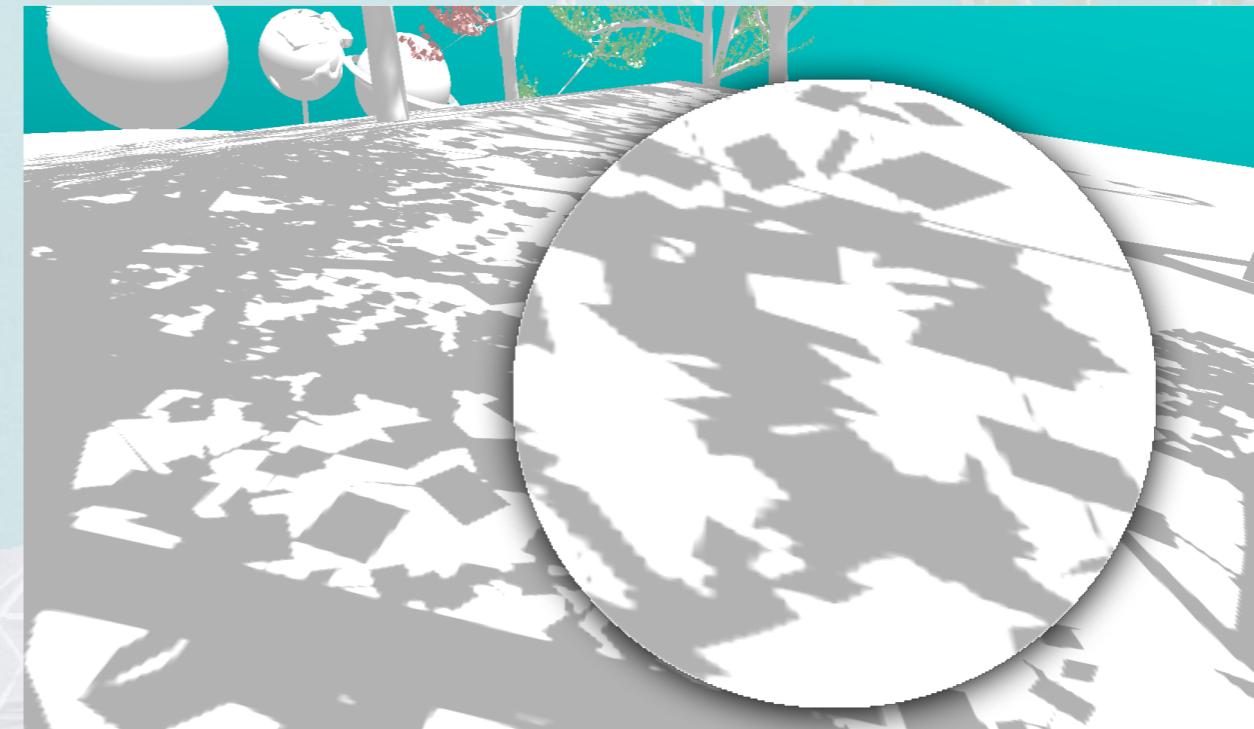
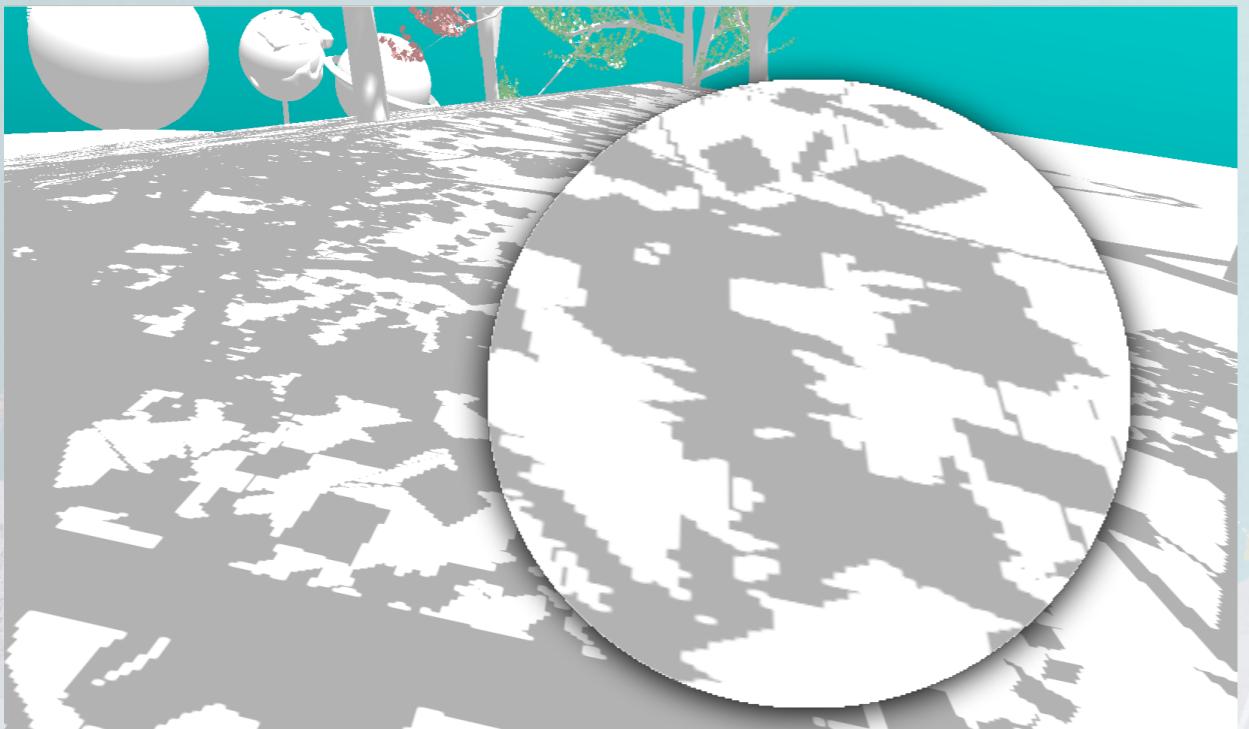
Szene mit Area Sampling



Szene ohne Area Sampling

4. MOMENT SHADOW MAPPING

- Echzeit Rendering & Hard Shadows mit hoher Qualität
- Anti-Aliasing(MSAA) auf Shadow Map
- nicht direkt filterbar, da sonst Geometrie verändert wird



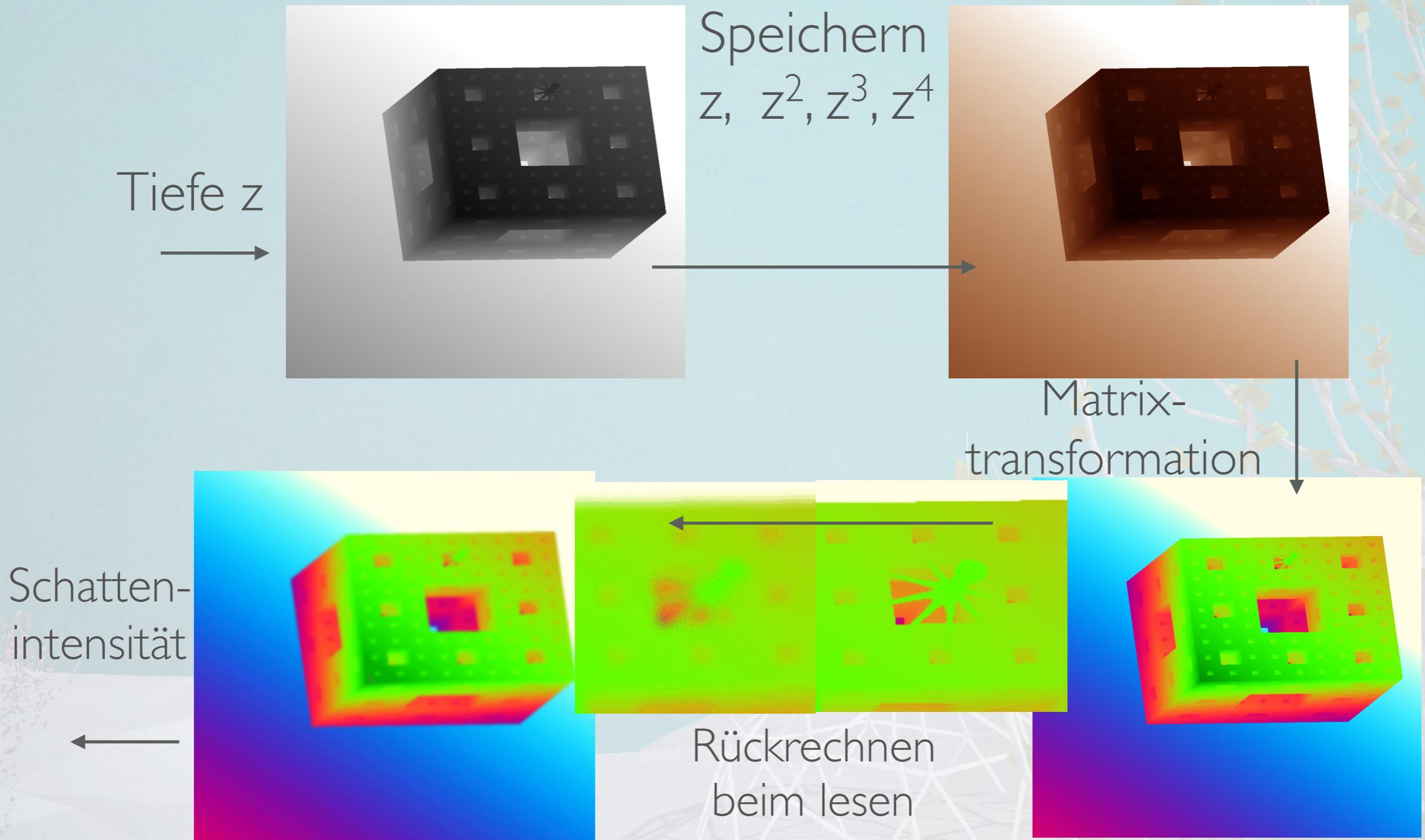
4. I MSM: ALGORITHMUS

- Hamburger Moment Shadow Mapping
- Tiefe und gefiltertes Sample der Moment Shadow Map, ähnlich Variance Shadow Maps
- Approximiert Schattenintensität
- Untere Schranke , Schattenintensität wird nicht überschätzt

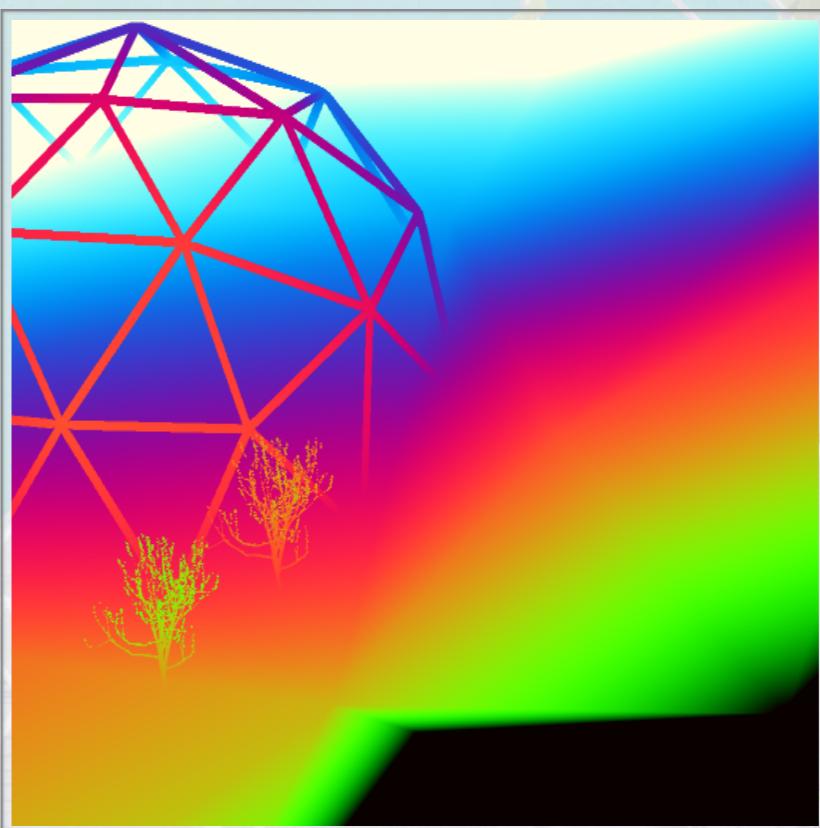
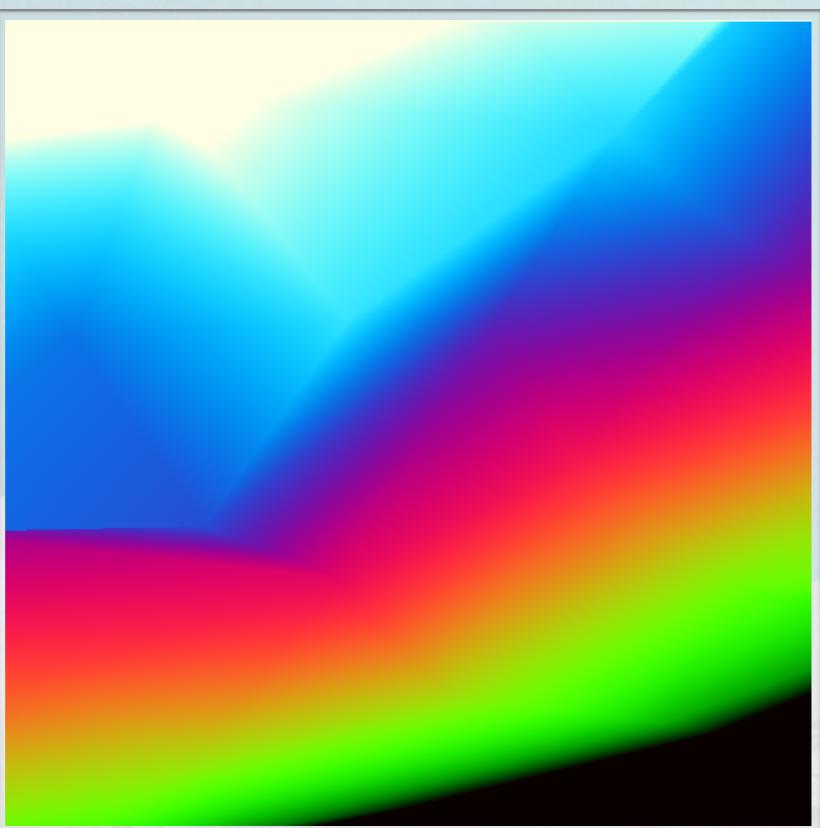
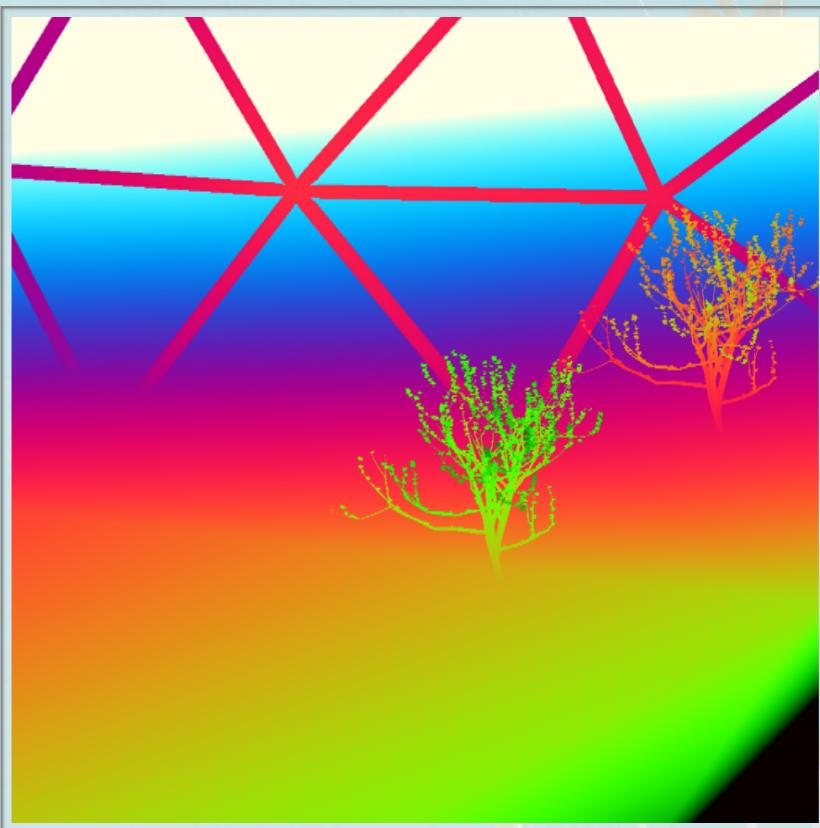
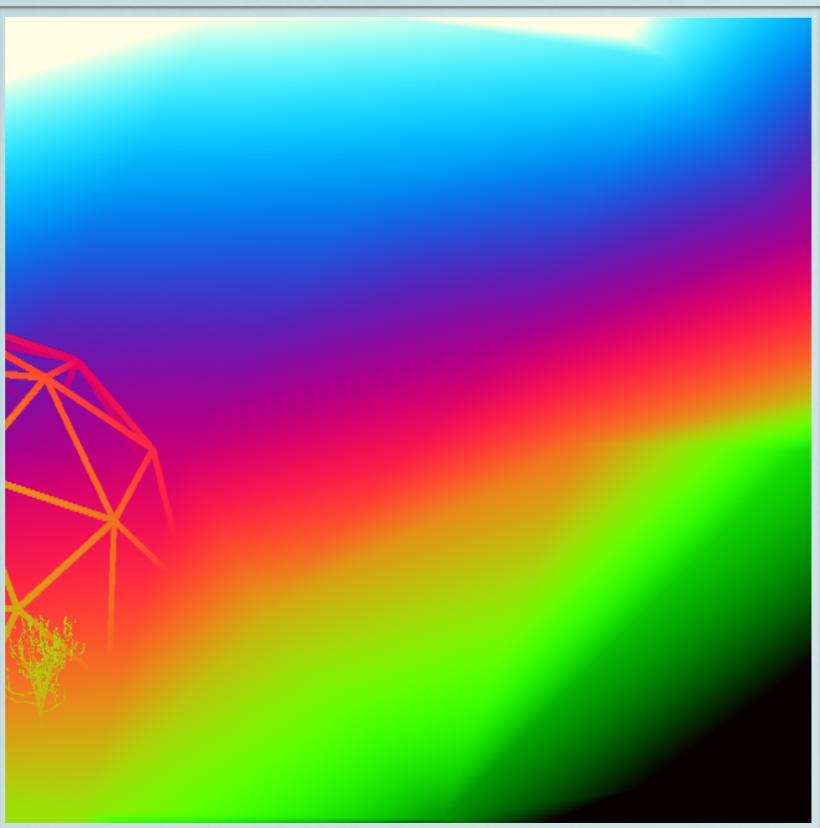
- Input: Depth $z_f \in \mathbb{R}$, moments $b_j = \mathbb{E}_Z(\mathbf{z}^j)$ for $j \in \{1,2,3,4\}$
- Output: Approximate shadow intensity

1. Solve $\begin{pmatrix} 1 & b_1 & b_2 \\ b_1 & b_2 & b_3 \\ b_2 & b_3 & b_4 \end{pmatrix} \cdot \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} 1 \\ z_f \\ z_f^2 \end{pmatrix}$
2. Solve $c_3 \cdot z^2 + c_2 \cdot z + c_1 = 0$ for $z_2, z_3 \in \mathbb{R}$
3. Solve $\begin{pmatrix} 1 & 1 & 1 \\ z_f & z_2 & z_3 \\ z_f^2 & z_2^2 & z_3^2 \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} 1 \\ b_1 \\ b_2 \end{pmatrix}$
4. Return $\sum_{i=2, z_i < z_f}^3 w_i$

4.2 MSM: IMPLEMENTIERUNG



4.3 MSM: MOMENT TEXTUR



4.4 MSM: ERGEBNISSE



4.4 MSM: ERGEBNISSE (2)

- Anti-Aliasing durch MSAA und Filtern durch Gauss
- Light Leaking wird besser verringert als bei VSM
- Kaum Auswirkungen auf Engine-Performance, das Filtern allerdings hohe Einbrüche

4.5 MSM: MBVO ALGORITHMUS

- Moment Based Volumetric Obscurrence
- Area Sampling uniforme Verteilung
- Nutzbar durch MSM Technik, deshalb robuster und effizient
- 3 Tiefen werden gewichtet aufsummiert für VO Term

$$\mathbf{f}(z) := \begin{cases} 1 & \text{if } z < z_a, \\ \frac{z - z_a}{z_b - z_a} & \text{if } z_a \leq z \leq z_b, \\ 0 & \text{if } z_b < z. \end{cases}$$

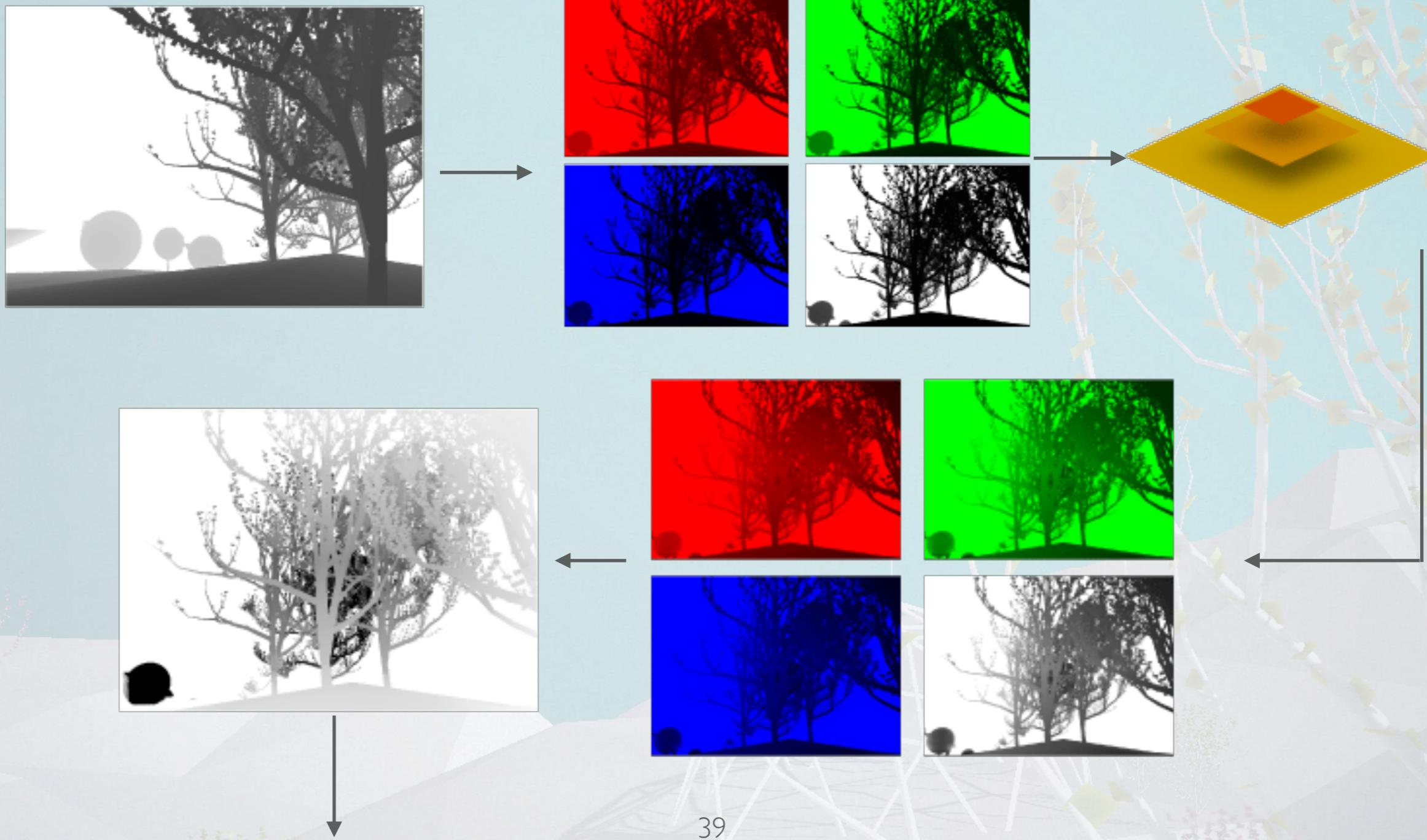


$$E_S(\mathbf{f}) = \sum_{i=1}^n w_i \cdot \mathbf{f}(z_i)$$

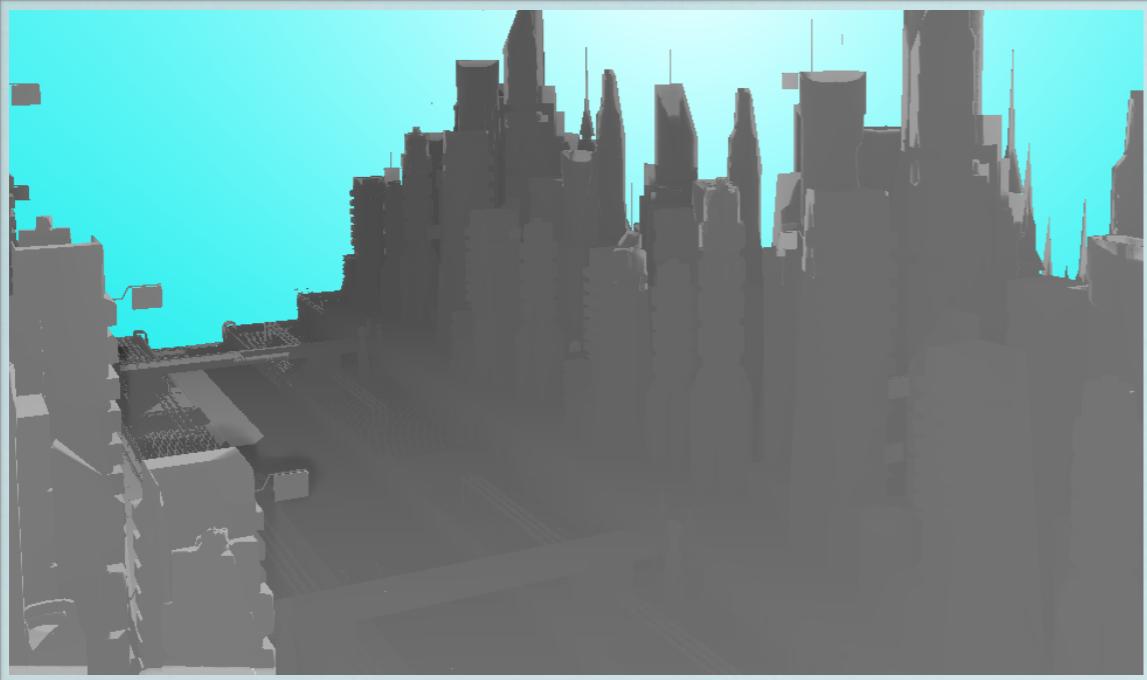
4.6 MBVO: IMPLEMENTIERUNG

- Moment Based Volumetric Obscurrence Term berechnen
- Ähnlicher Aufbau wie für MSM
- Funktion der Intensität wird erweitert um Gewichte
- Summe aus Gewichten und $F(z)$ Werten

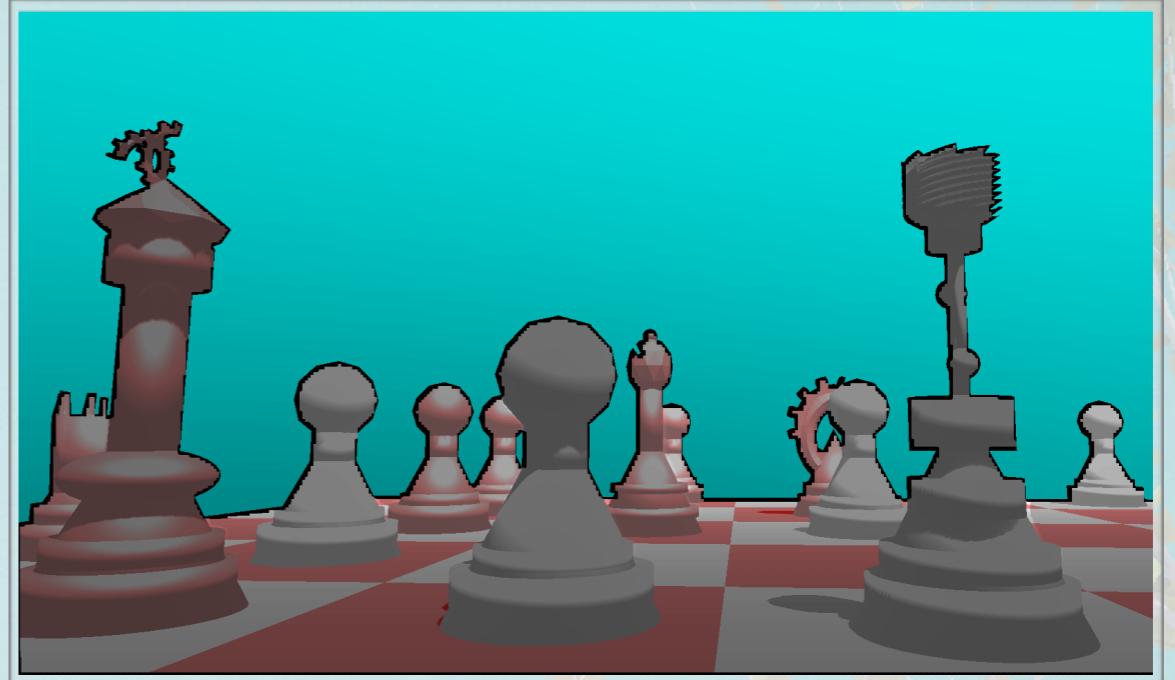
4.6 MBVO: IMPLEMENTIERUNG



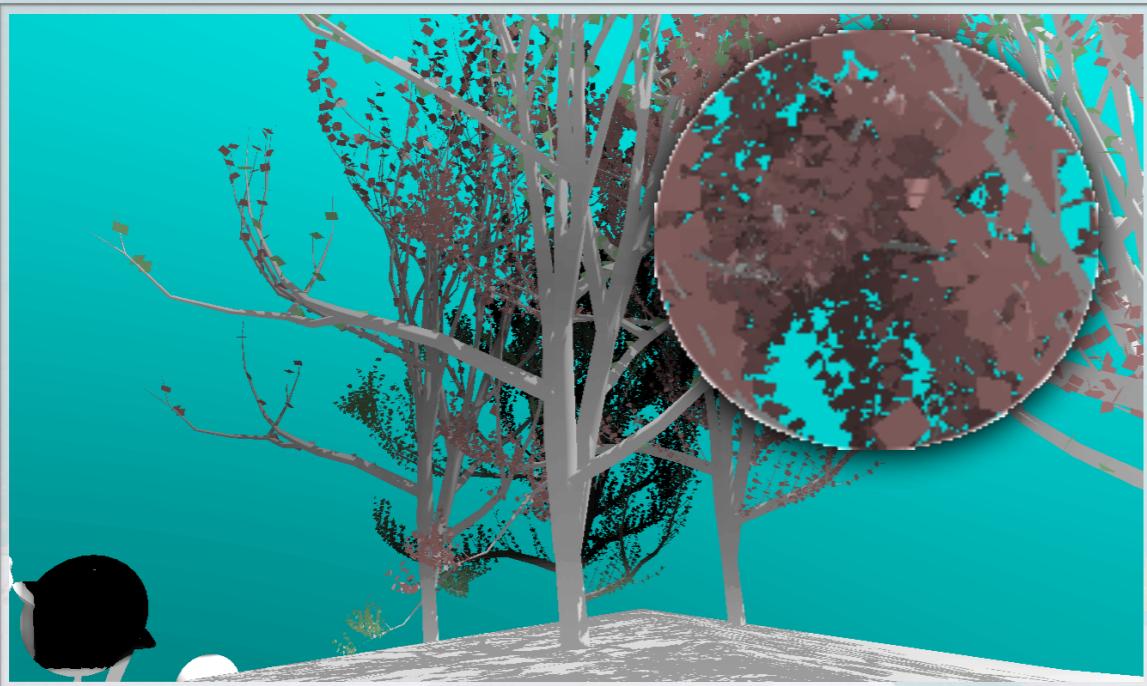
4.7 MBVO: ERGEBNISSE



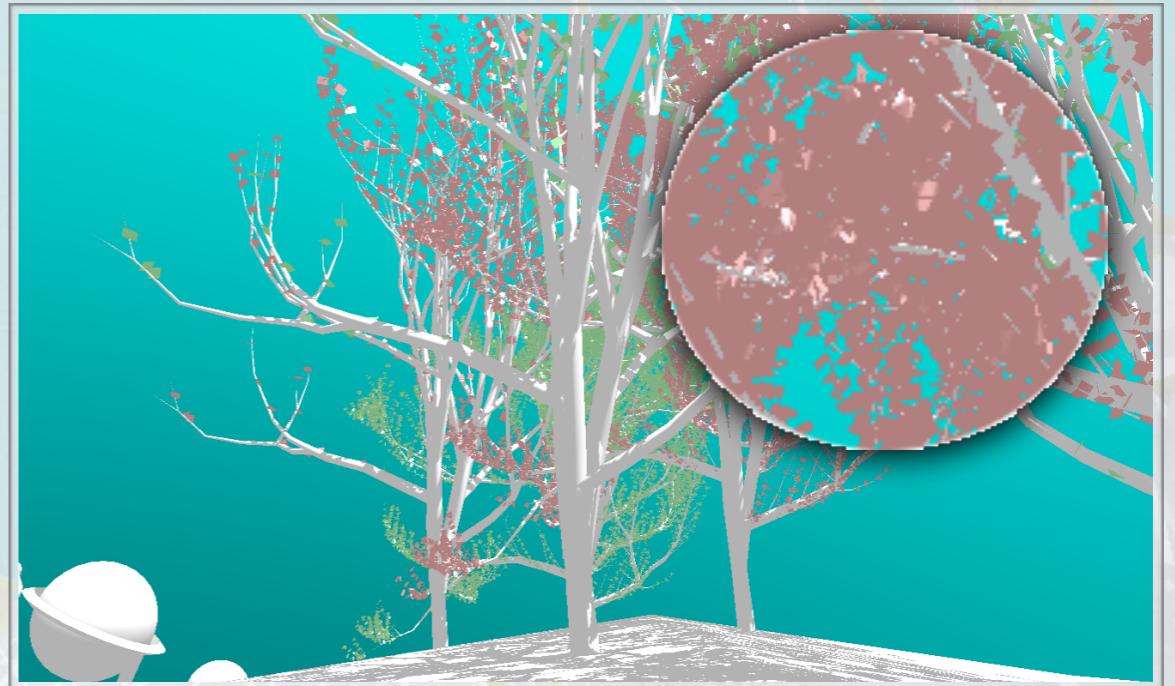
MBVO I



MBVO 2



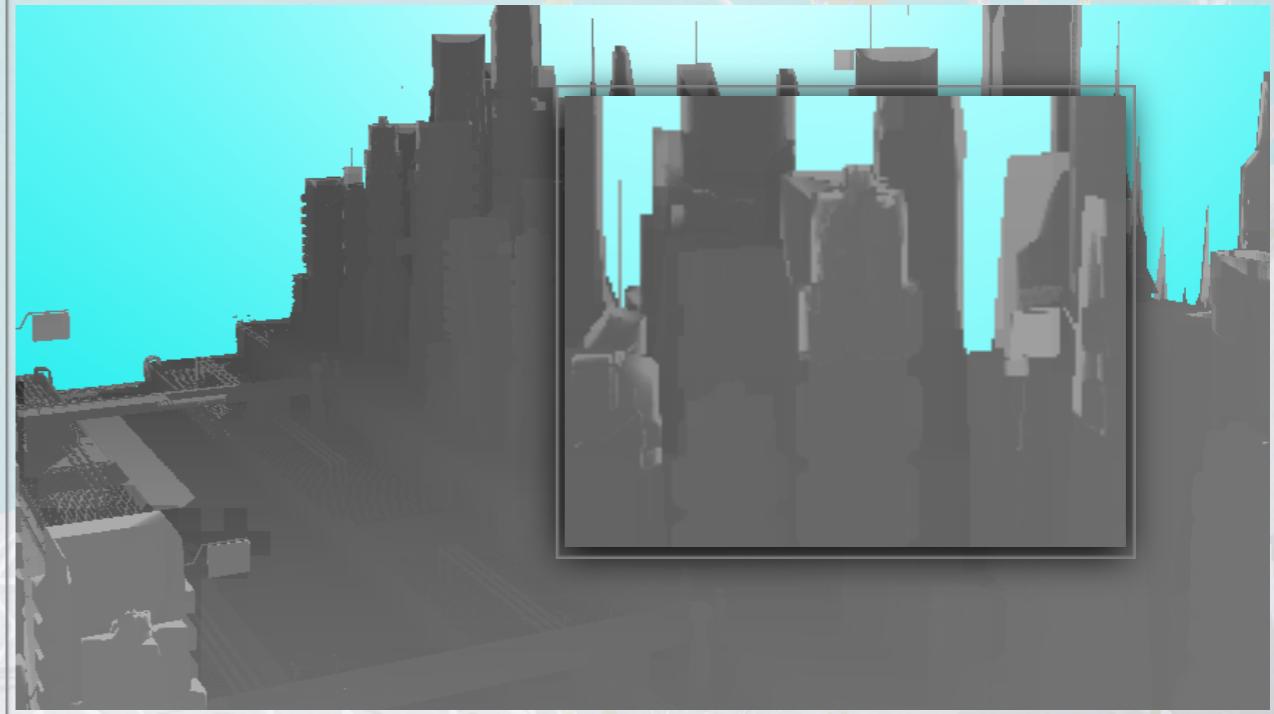
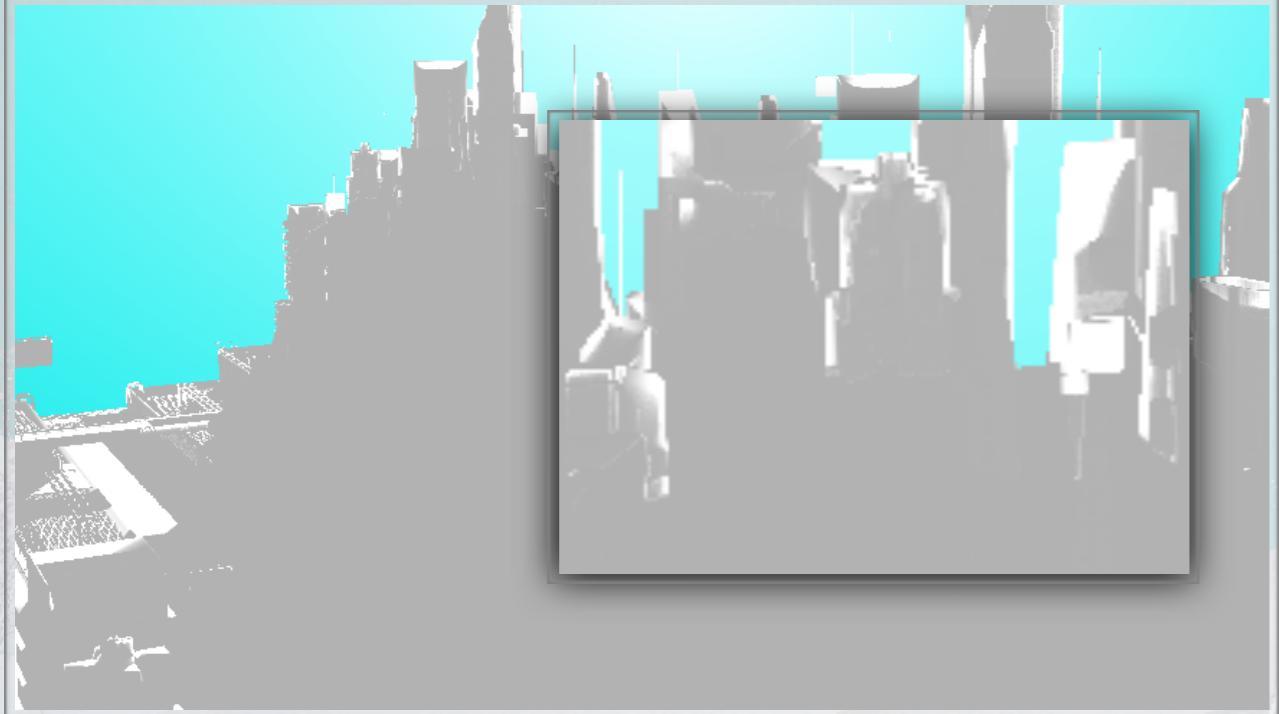
MBVO ON



MBVO OFF

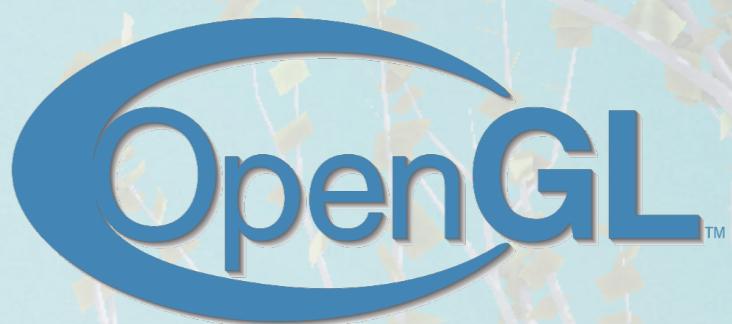
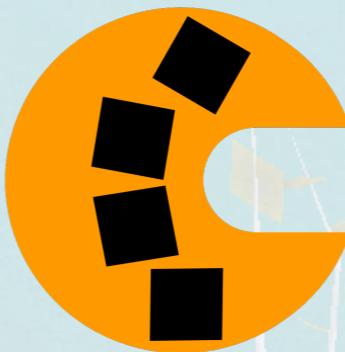
4.7 MBVO: ERGEBNISSE (2)

- Leicht umsetzbar durch Moment Technik
- Cell Shading ähnlicher Effekt mit falschen Werten
- Weiteres experimentieren nötig um ästhetisches Bild zu erhalten
- `glGenerateMipmaps`

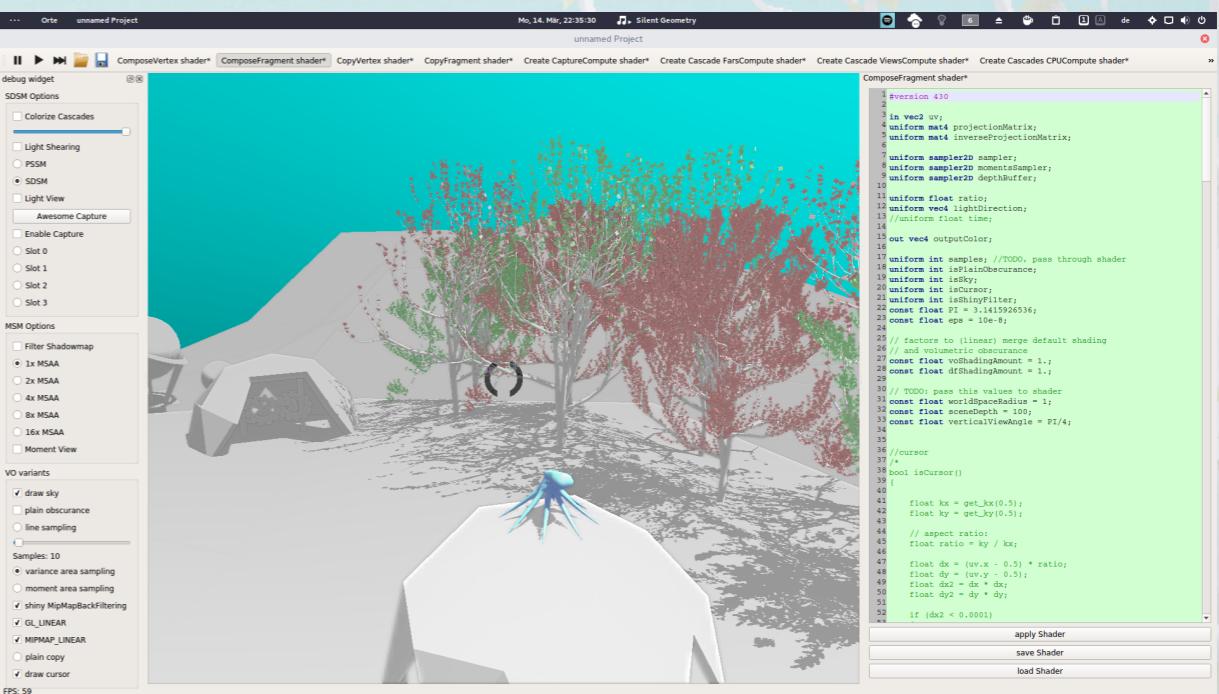
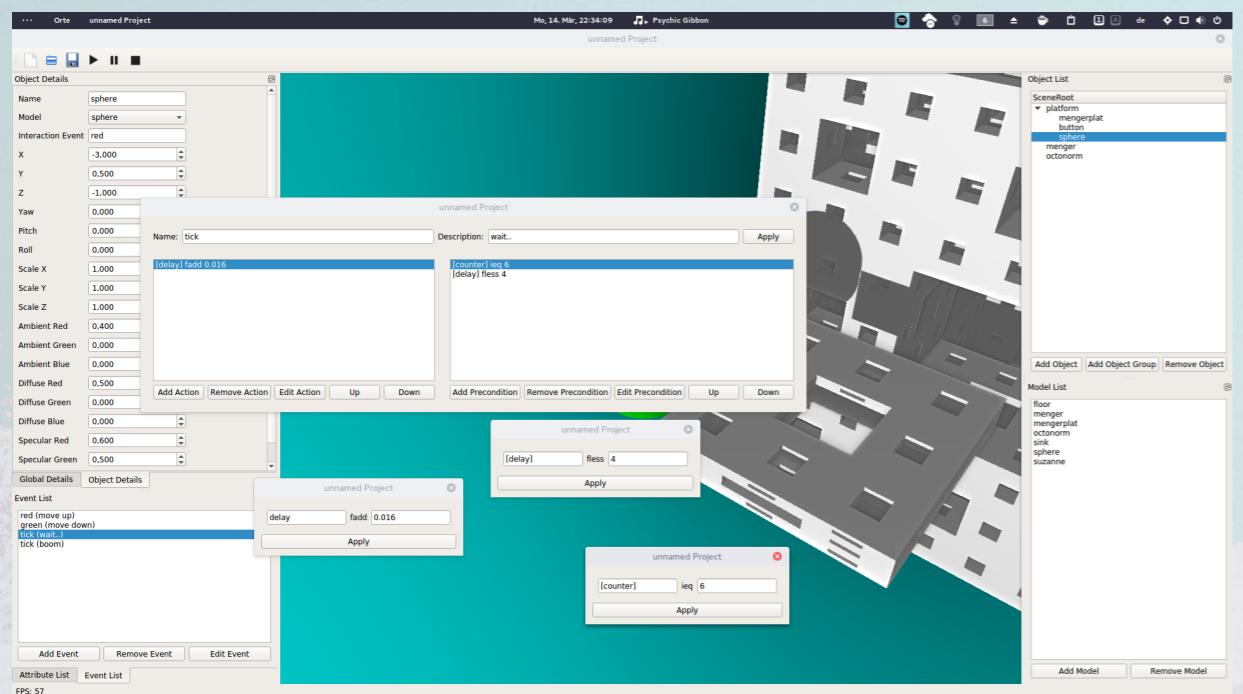
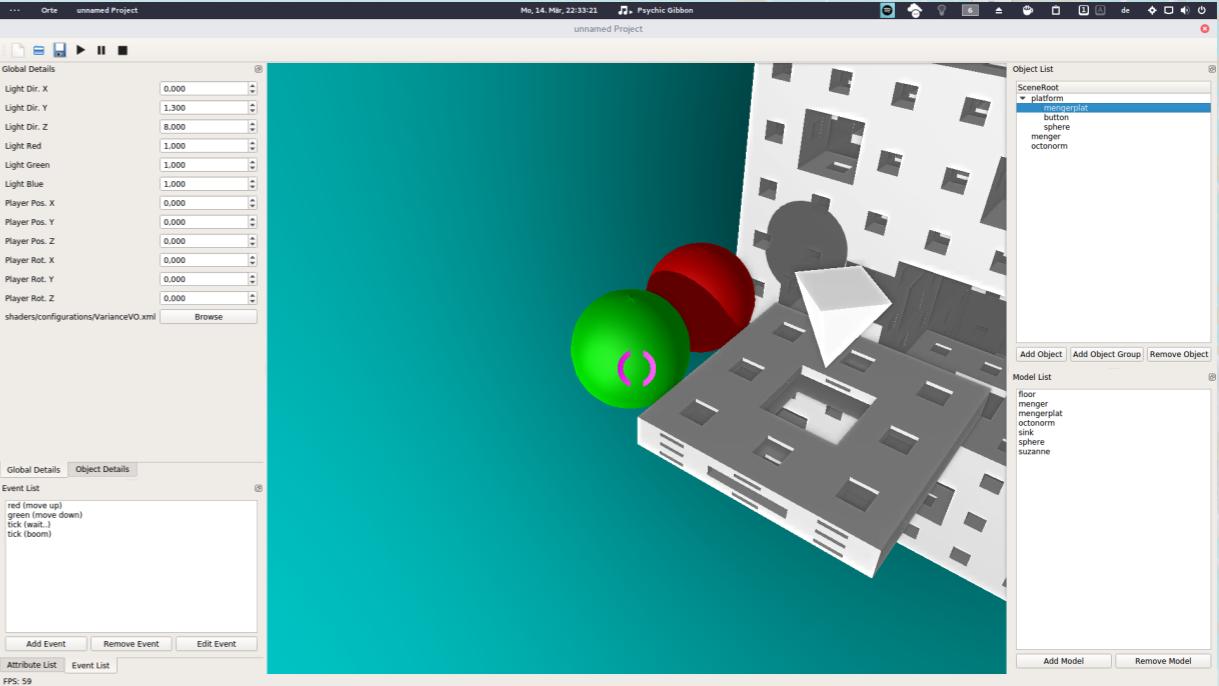
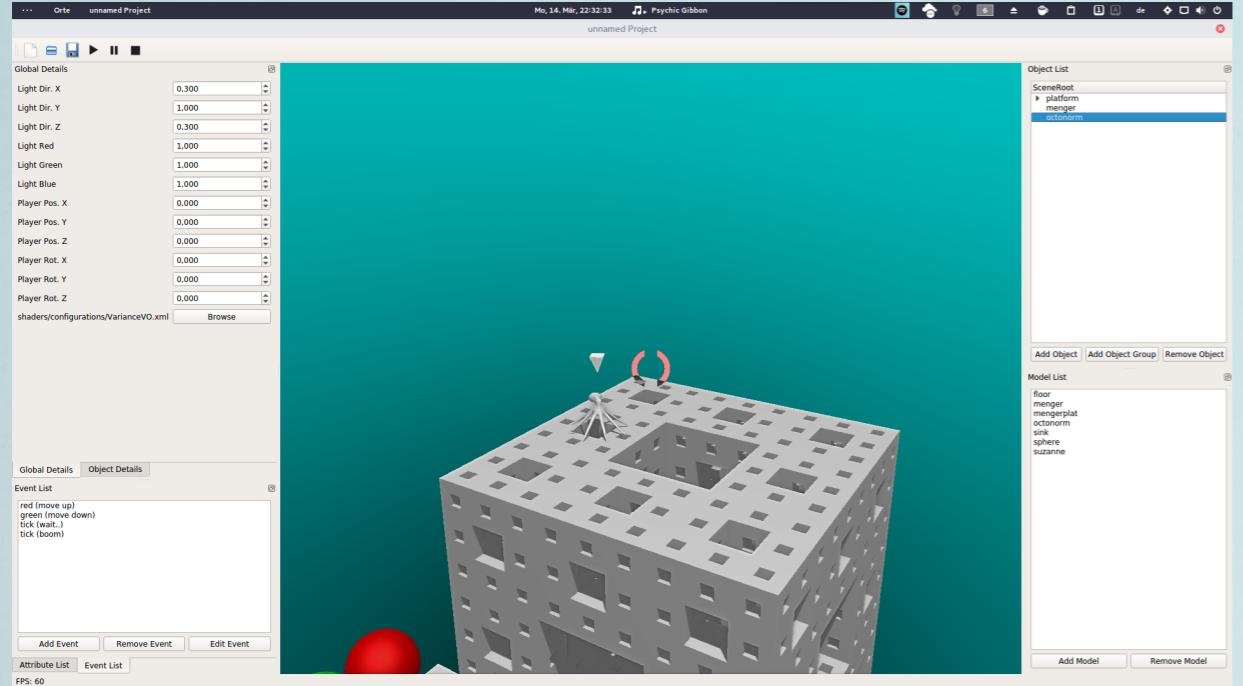


5. SUSHI! WITH R.I.C.E.

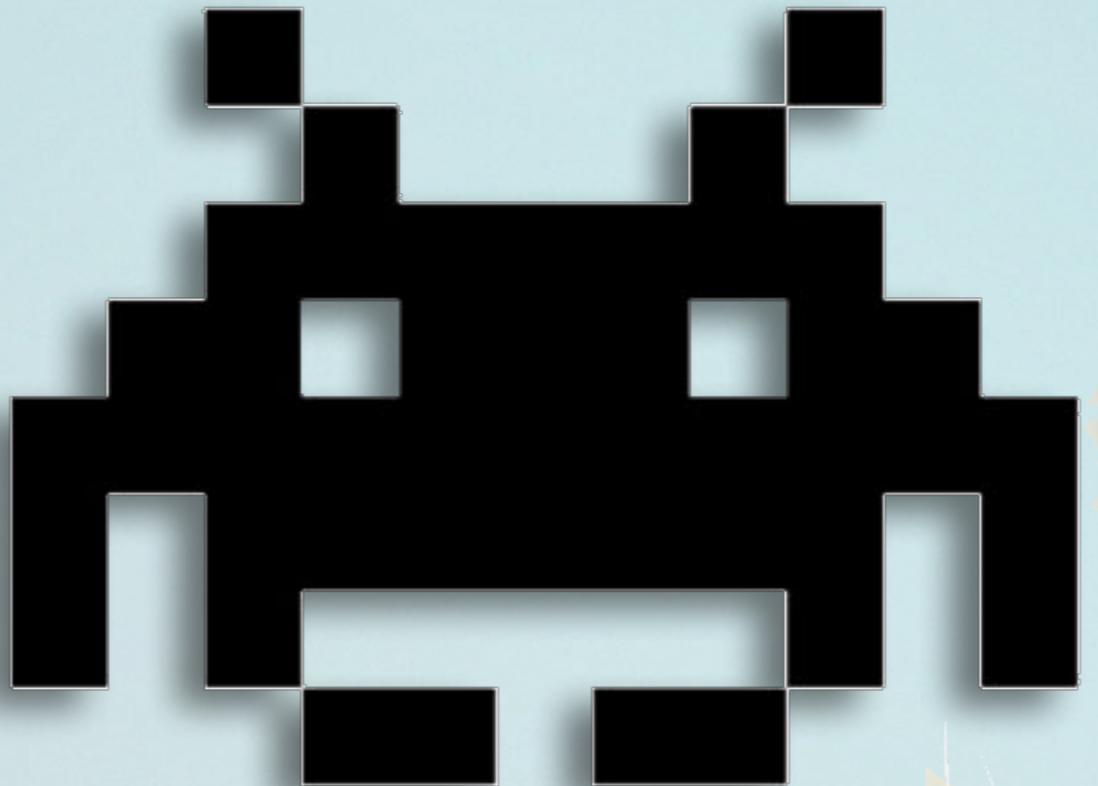
- Realistic Insane Computer-Graphic Environment
- C++, Qt, OpenGL, Bullet Physics
- Linux und Windows Support
- Sushi! Game Demo



5. ENTWICKLERTOOLS



5. I SUSHI!

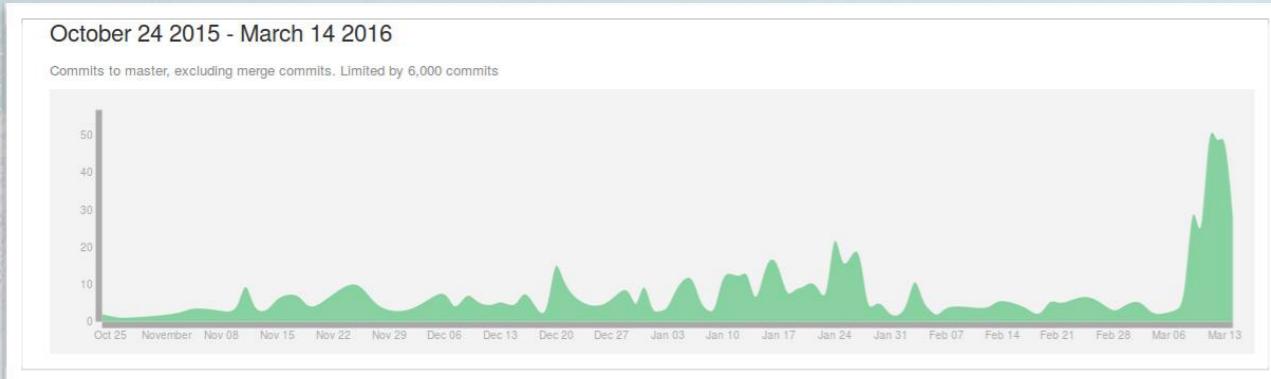


PRESS START



6. FAZIT UND AUSSICHTEN

- Spiel soll fertig entwickelt werden
 - weitere Techniken und Verbesserungen
 - weitere Levels, Rätsel, Menüs und Optionen
- Planung vs. Chaos
- Wir haben viel gelernt
- Wir hatten jede Menge Spaß



VIELEN DANK
FÜR DIE
AUFMERKSAMKEIT