17. Előadás: Gaussian Splatting

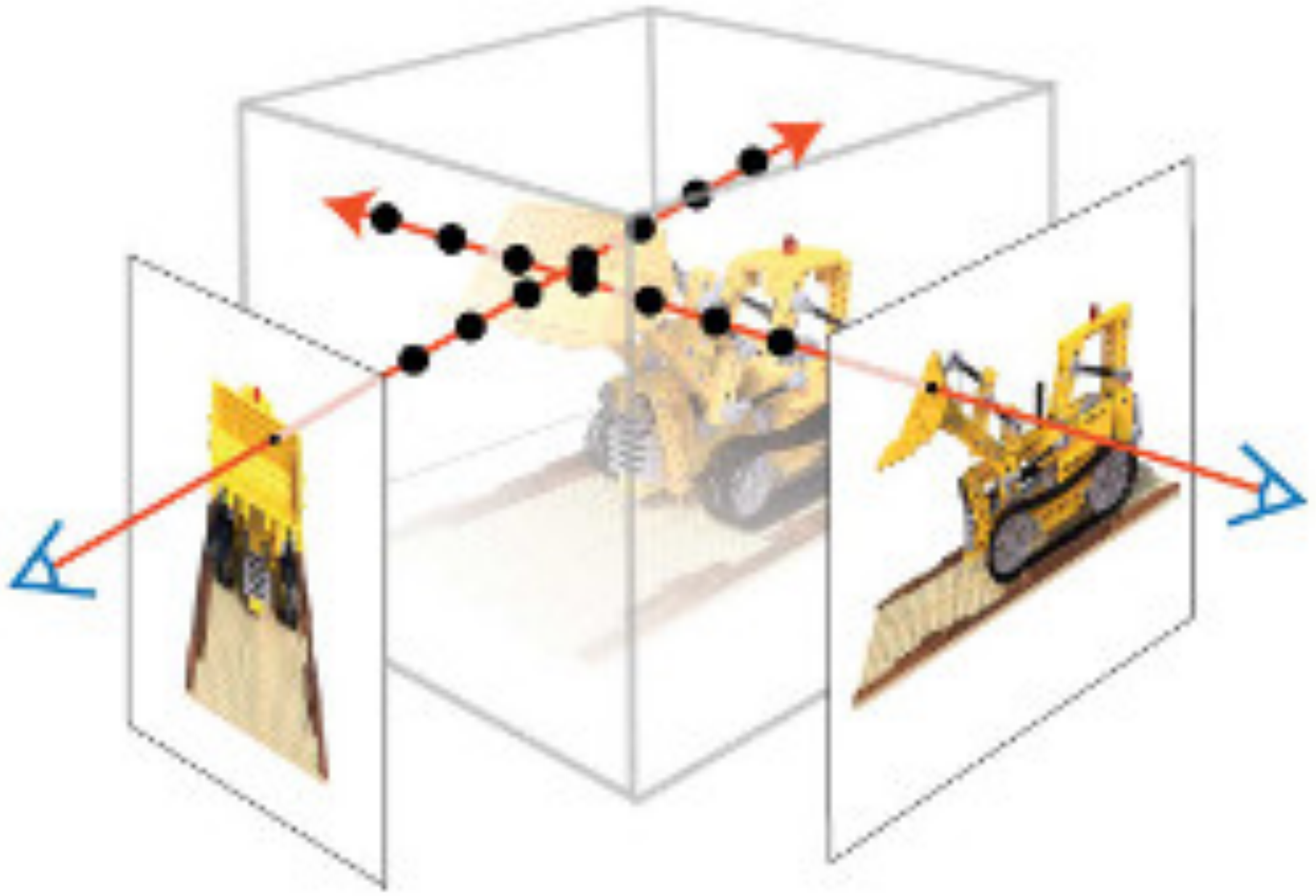
Generatív AI és Inverz Módszerek a Képszintézisben

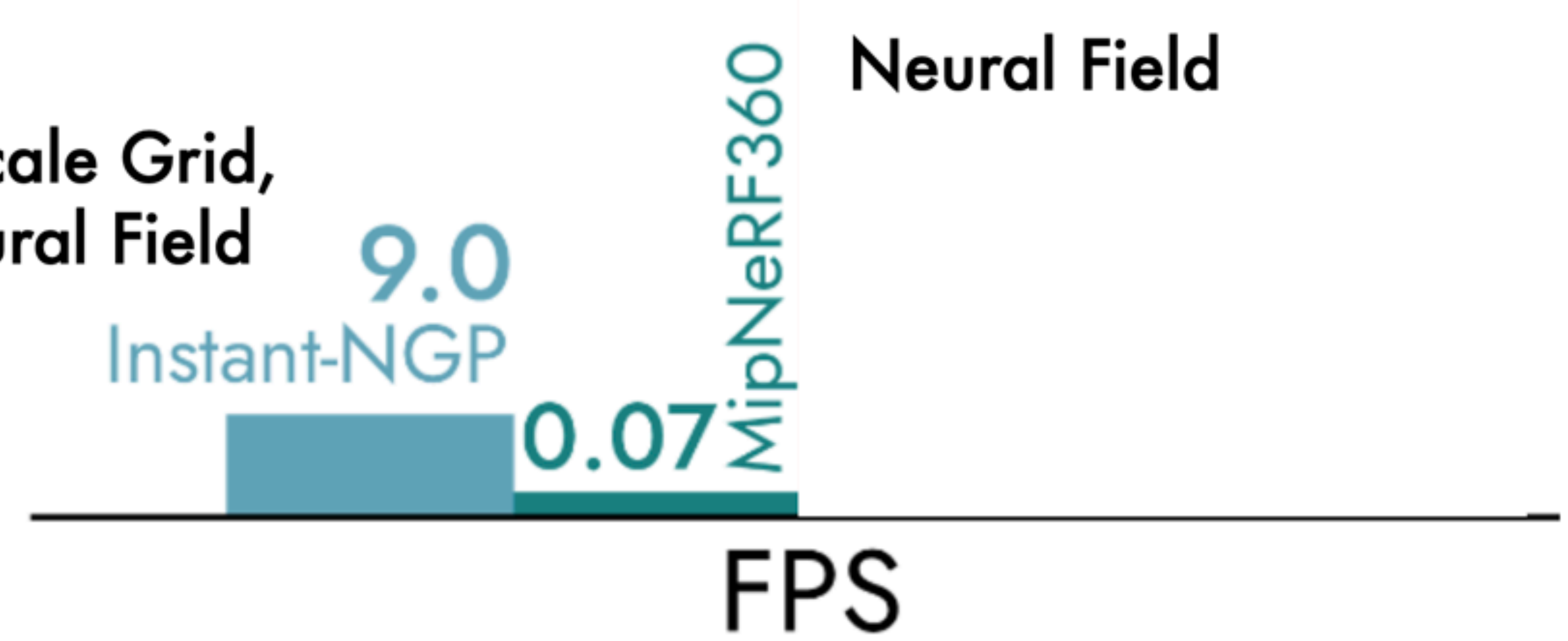
BME-VIK IIT, 2026

Dr. Vaitkus Márton

Radiancia Mezők

Volume rendering — van más út?



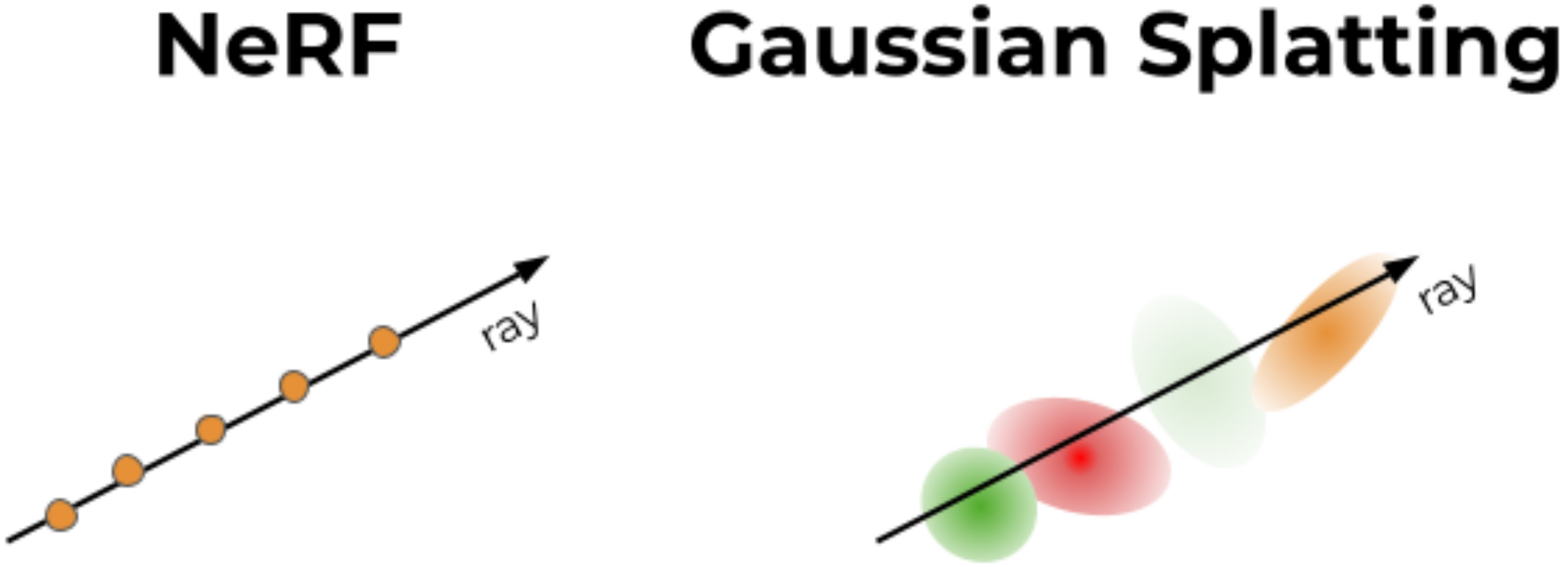


NeRF, stb. — implicit reprezentáció & volume rendering

Jó minőség, de lassú rendering!

Radiancia Mezők

Volume rendering — van más út?

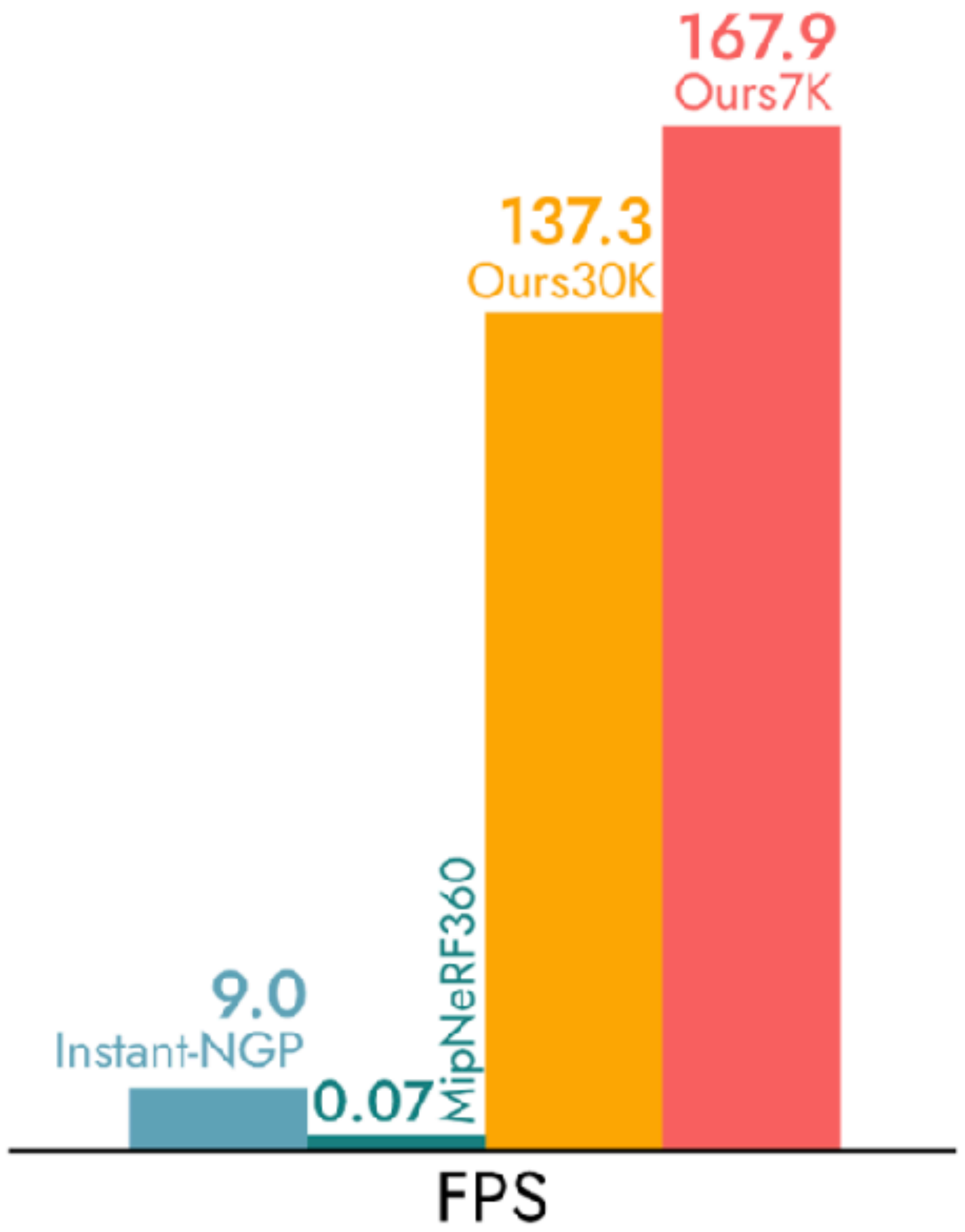


<u>**3D Gaussian Splatting (3DGS)**</u>: *explicit* reprezentáció + *raszterizáció*
jó minőség & <u>gyors rendering</u>

3D Gaussian Splatting

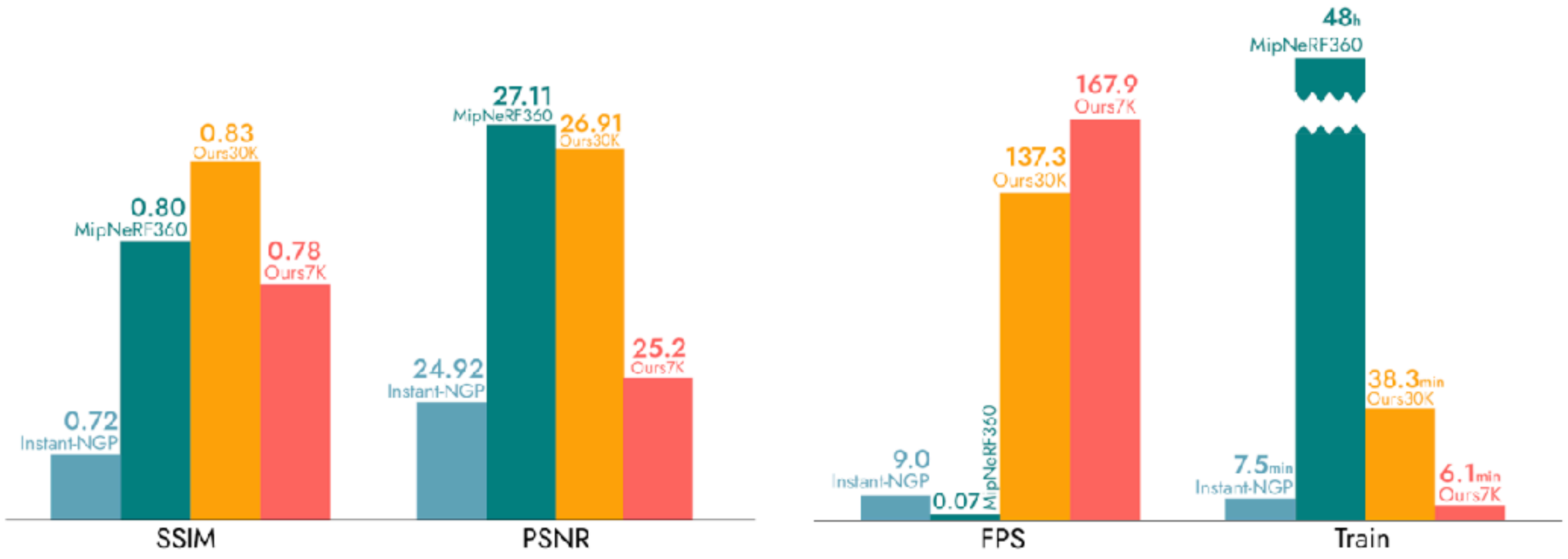
3D Gaussian Splatting for Real-Time Radiance Field Rendering

BERNHARD KERBL*, Inria, Université Côte d'Azur, France
GEORGIOS KOPANAS*, Inria, Université Côte d'Azur, France
THOMAS LEIMKÜHLER, Max-Planck-Institut für Informatik, Germany
GEORGE DRETTAKIS, Inria, Université Côte d'Azur, France



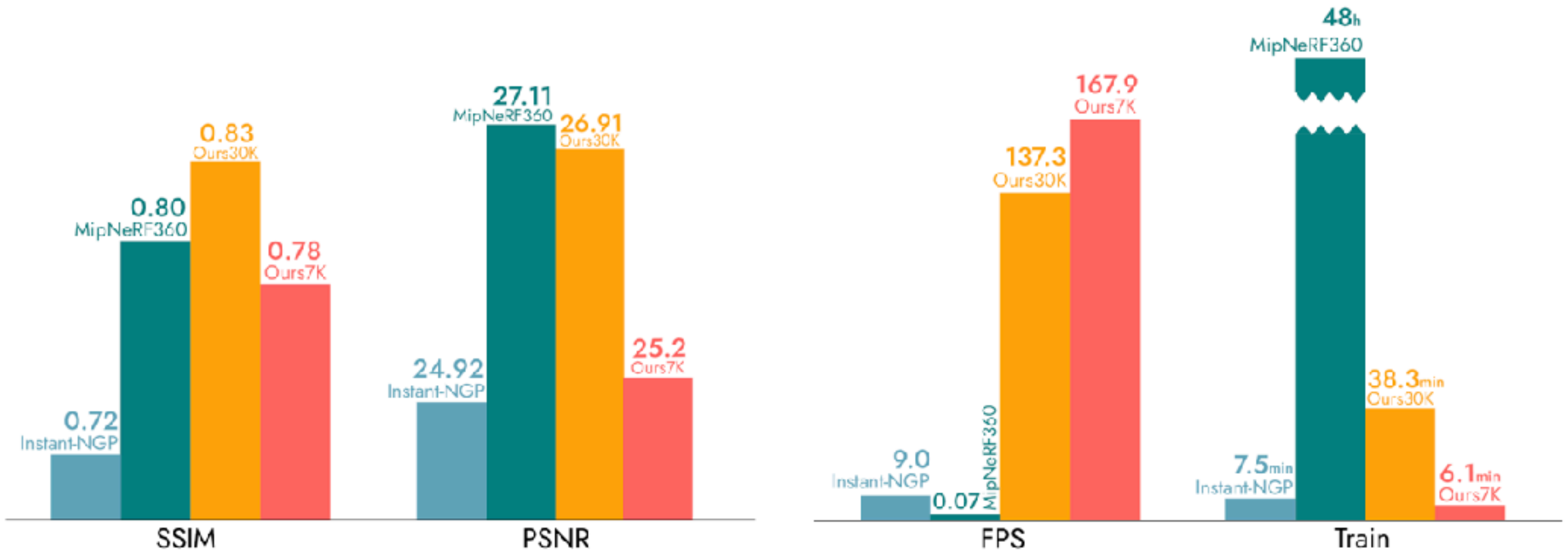
<u>3D Gaussian Splatting (3DGS)</u>: *explicit* reprezentáció + *raszterizáció*
jó minőség & <u>gyors rendering</u>

3D Gaussian Splatting



https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

3D Gaussian Splatting



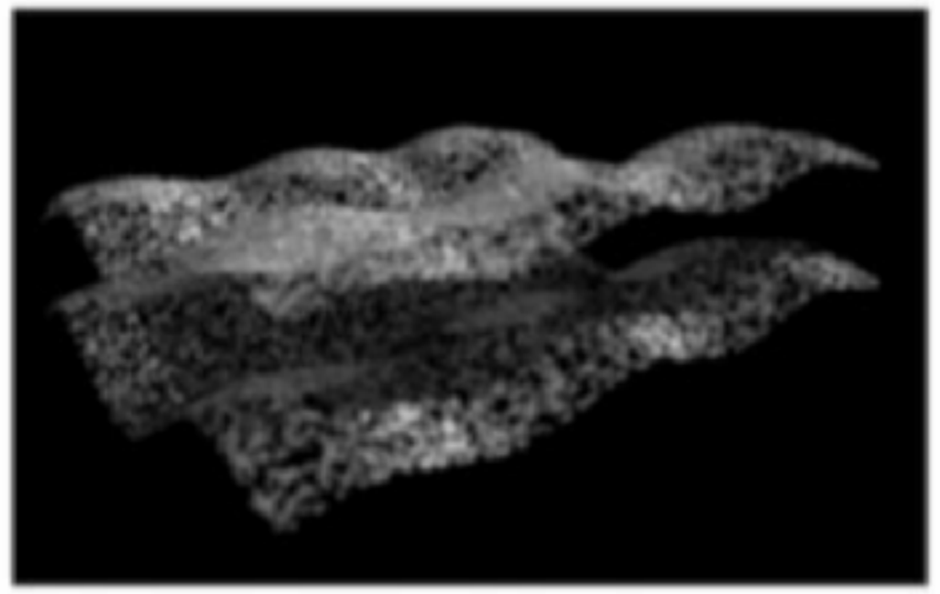
https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

3D Gaussian Splatting

Előzmények

The Use of Points as a Display Primitive

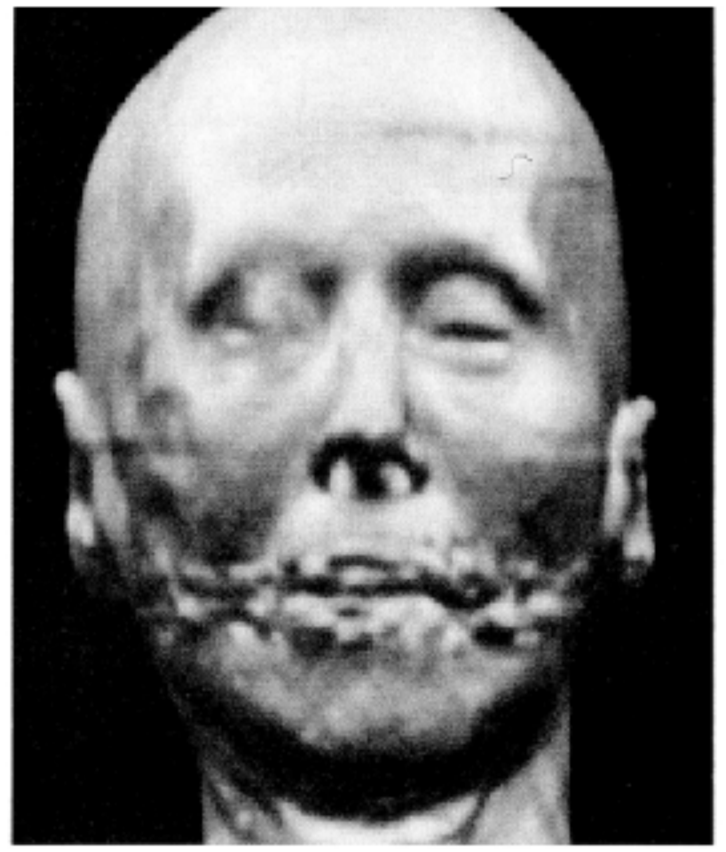
Marc Levoy
Turner Whitted

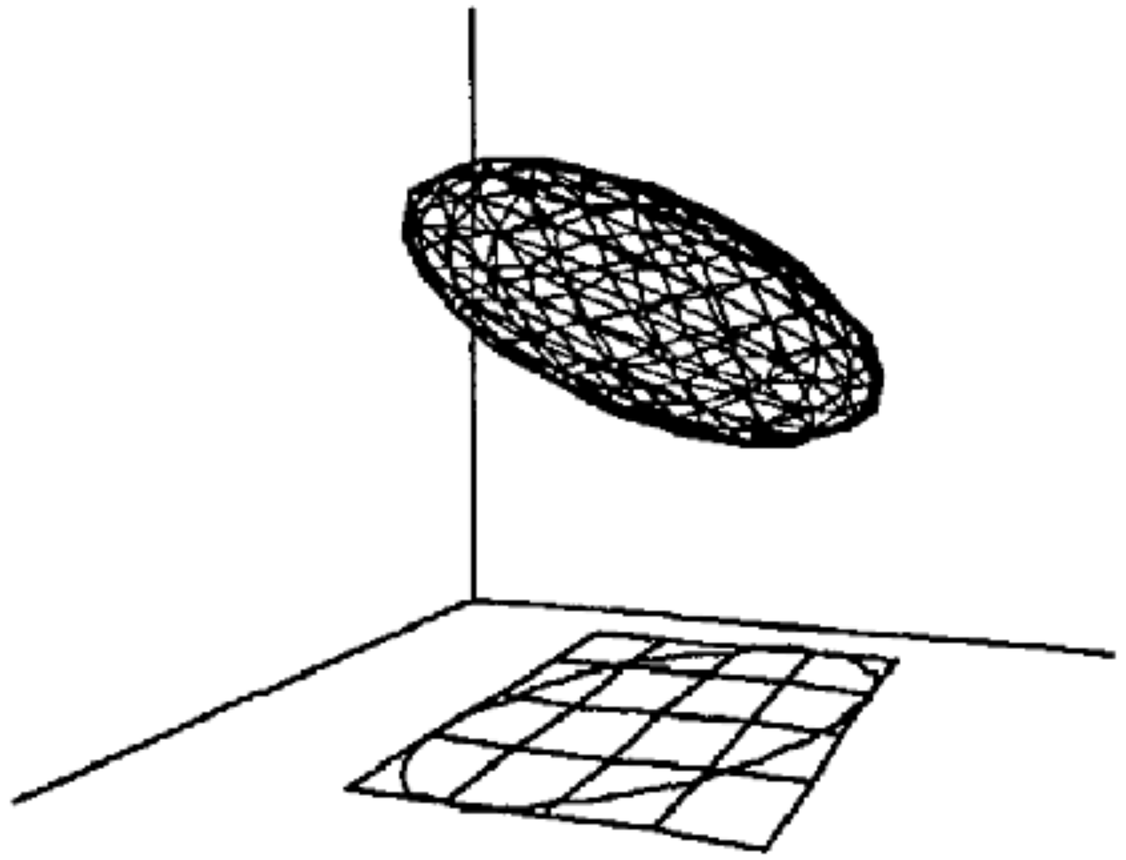


[1985]

SPLATTING:
A Parallel, Feed-Forward Volume Rendering Algorithm

Lee Alan Westover





[1990]

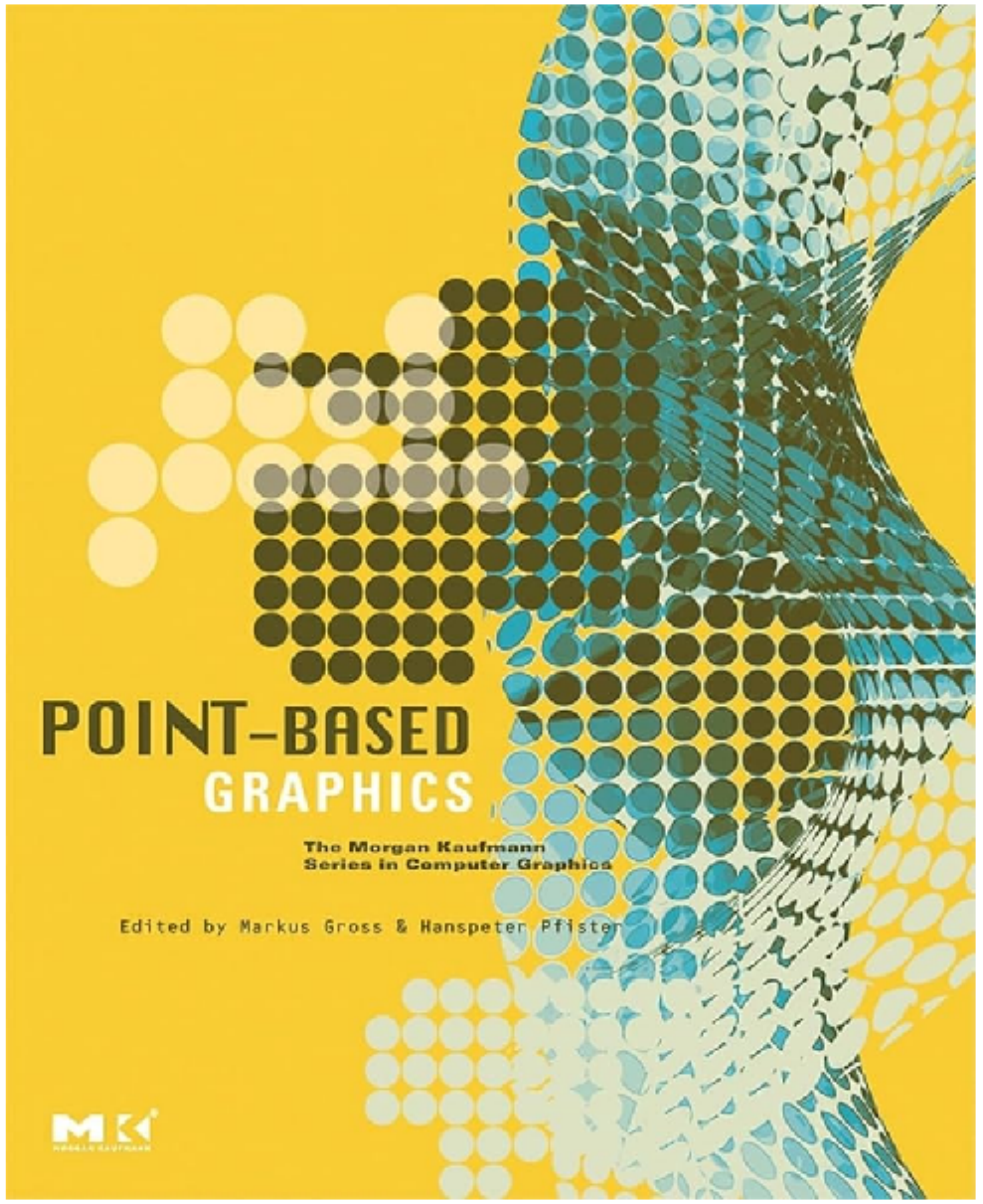
3D Gaussian Splatting

Előzmények


IEEE TRANSACTIONS ON VISUALIZATION AND COMPUTER GRAPHICS, VOL. 8, NO. 3, JULY-SEPTEMBER 2002

EWA Splatting

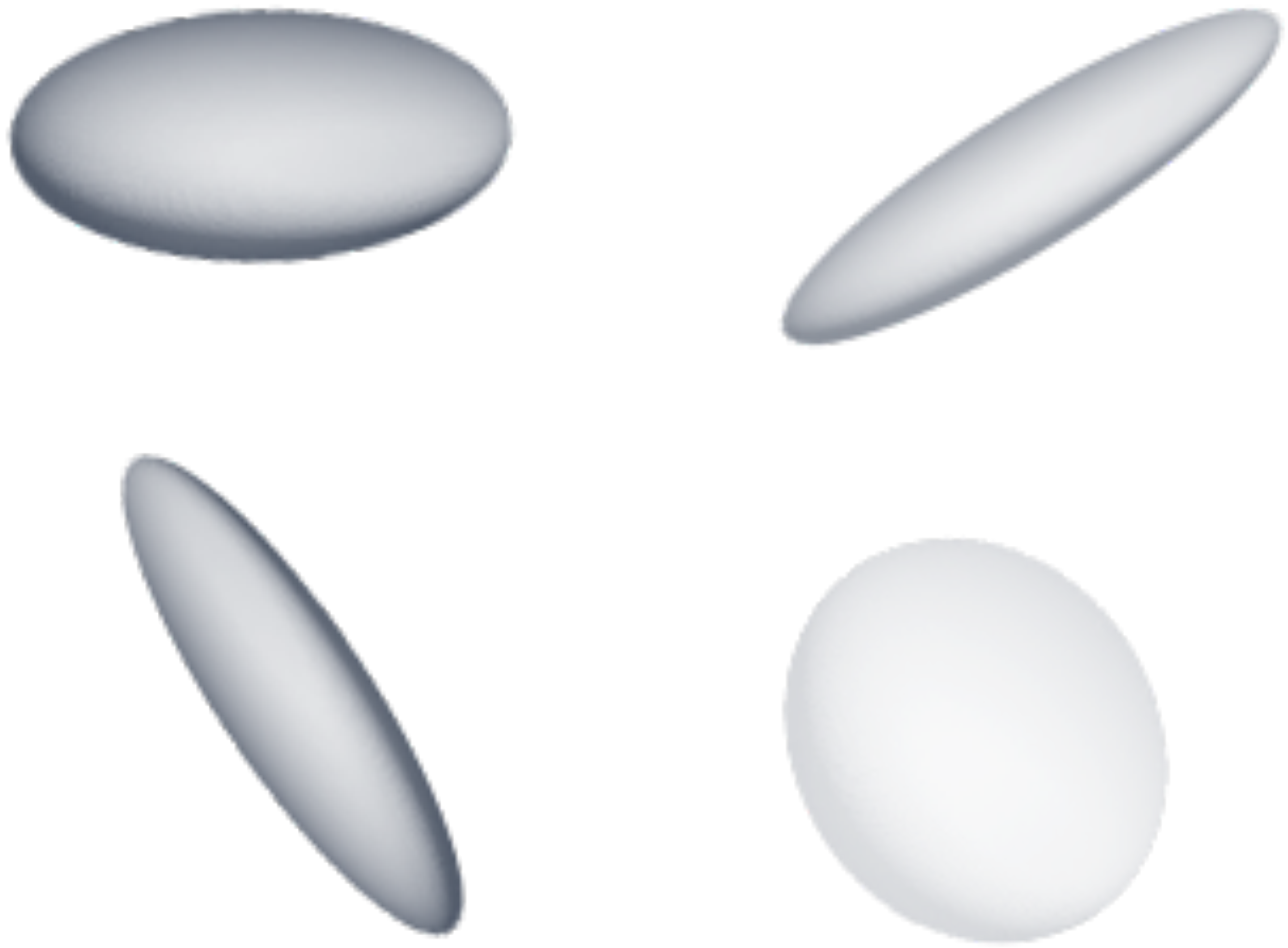
Matthias Zwicker, Hanspeter Pfister, *Member, IEEE*,
Jeroen van Baar, and Markus Gross, *Member, IEEE*



<u>Pont-alapú grafika</u> — A ’00-s évek divatos kutatási témája…

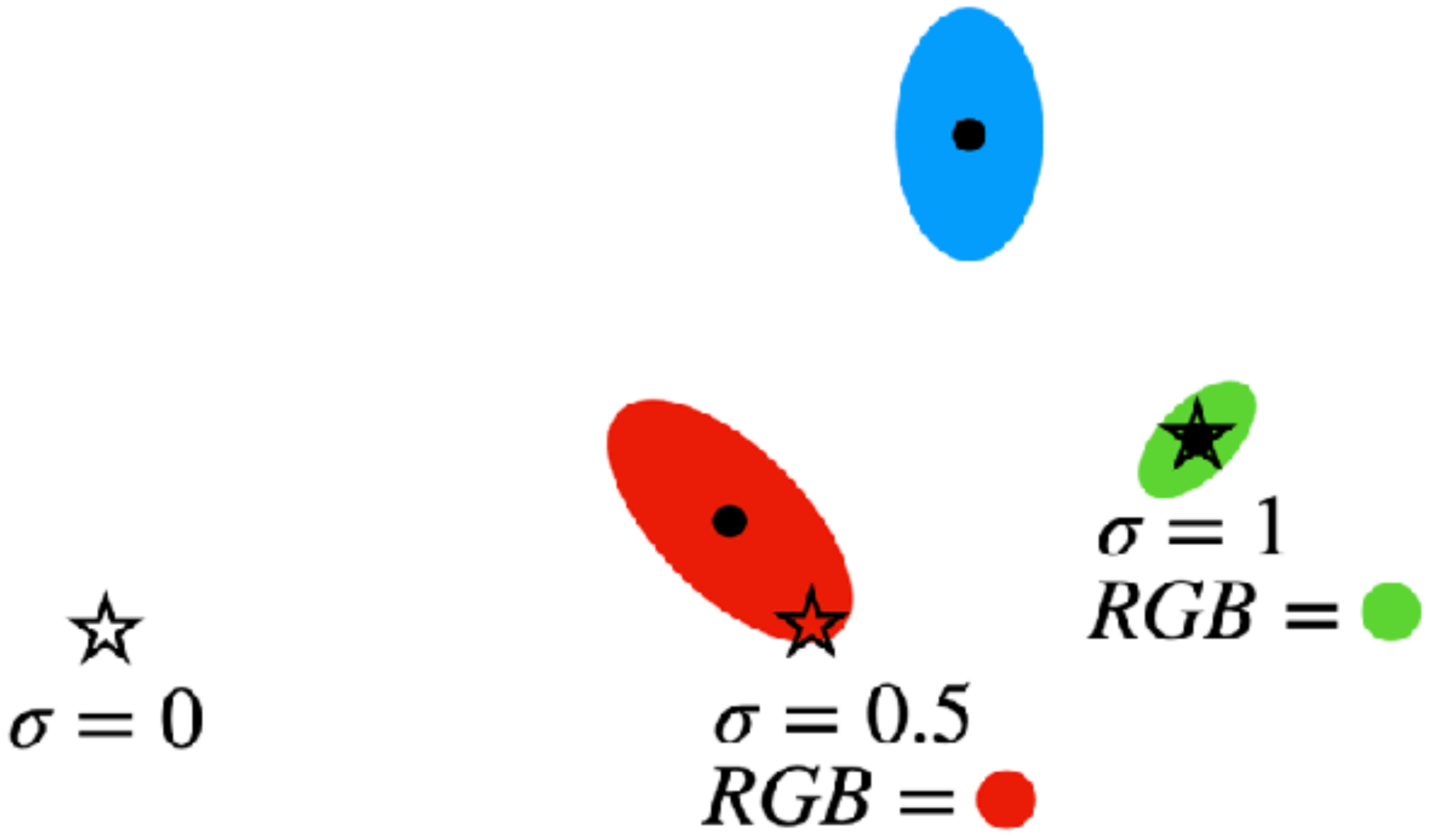
3D Gaussian Splatting

Ötlet: reprezentáljuk a radiancia mezőt explicit módon a térben
3D pontok, ellipszoid alakú “kiterjedéssel”
3D Gauss szűrővel “szétkent” sűrűség

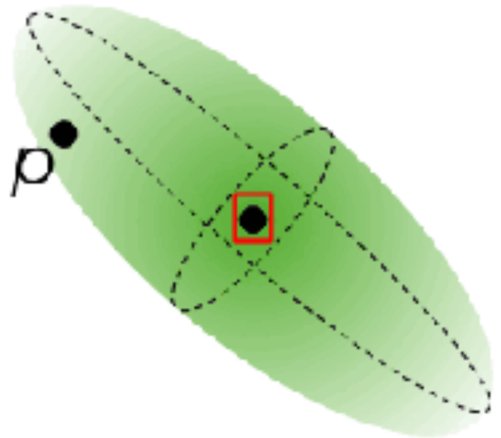


3D Gaussian Splatting

Ötlet: reprezentáljuk a radiancia mezőt explicit módon a térben
3D pontok, ellipszoid alakú “kiterjedéssel”
3D Gauss szűrővel “szétkent” sűrűség



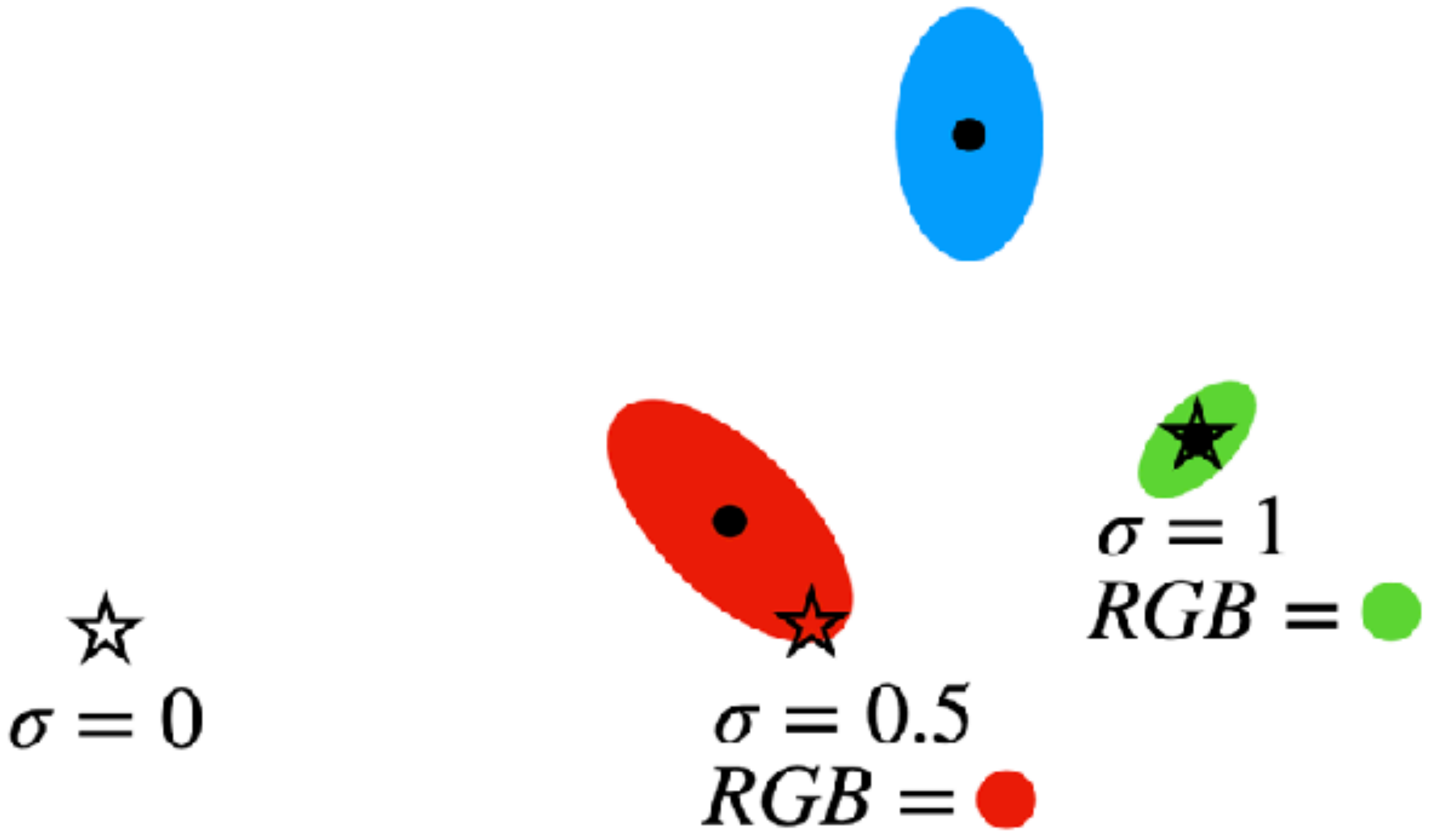
$$f_i(p) = \sigma(\alpha_i)\exp\left(-\frac{1}{2}(p - \boxed{\mu_i})\Sigma_i^{-1}(p - \boxed{\mu_i})\right)$$



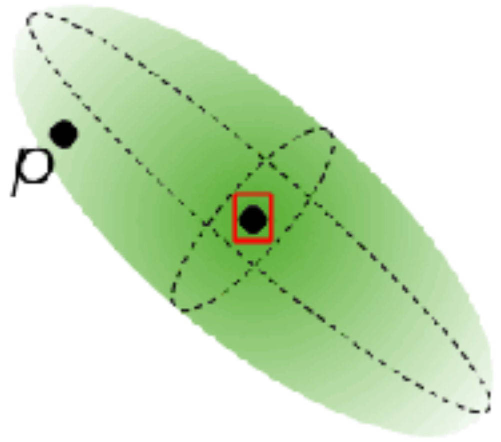
Slide credit: Vincent Sitzmann

3D Gaussian Splatting

Ötlet: reprezentáljuk a radiancia mezőt explicit módon a térben
3D pontok, ellipszoid alakú “kiterjedéssel”
3D Gauss szűrővel “szétkent” sűrűség

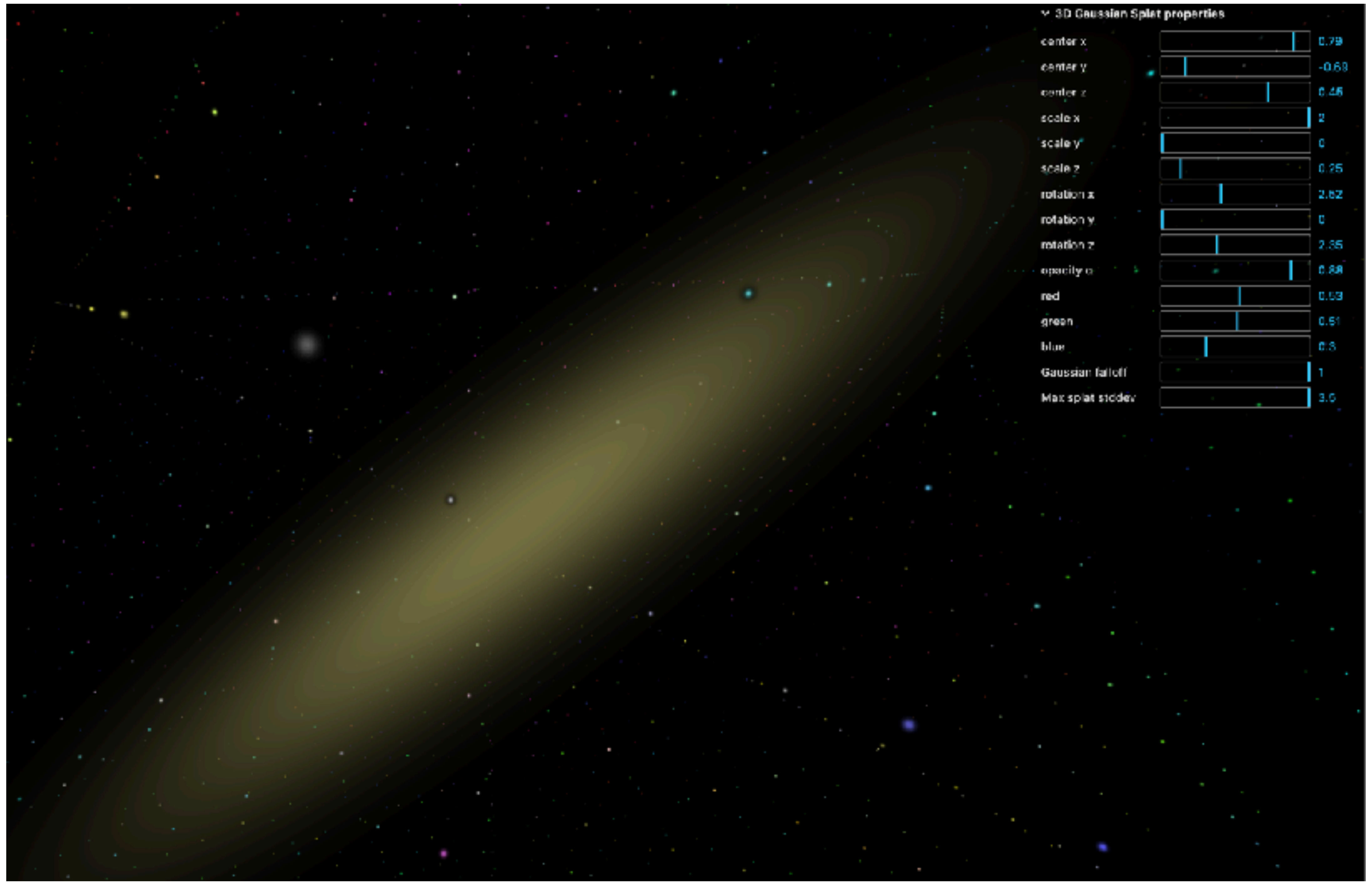


$$f_i(p) = \sigma(\alpha_i)\exp(-\frac{1}{2}(p-\boxed{\mu_i})\Sigma_i^{-1}(p-\boxed{\mu_i}))$$



Slide credit: Vincent Sitzmann

3D Gaussian Splatting

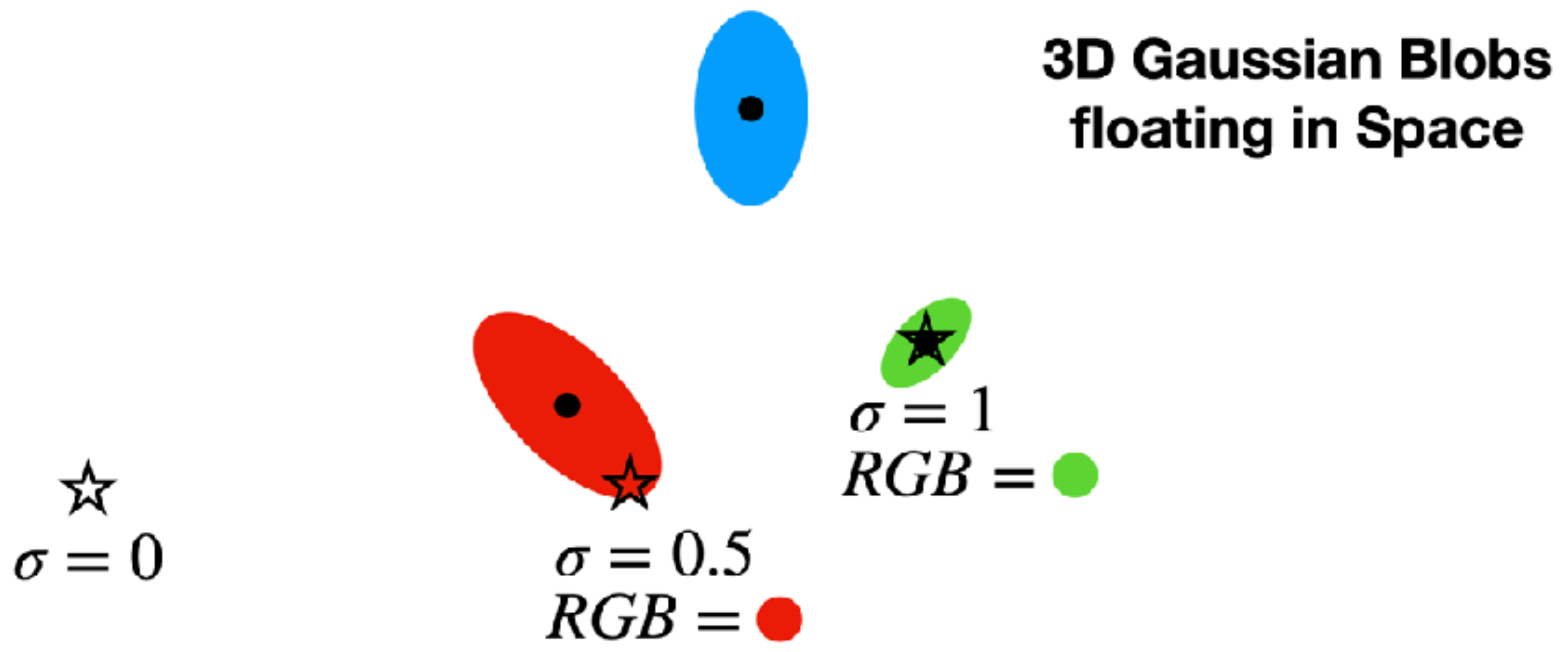


https://wlt-ai-cdn.art/spark-2.0/260413/3dgs-props.html

3D Gaussian Splatting

Rendering?

Hogyan rendereljük?

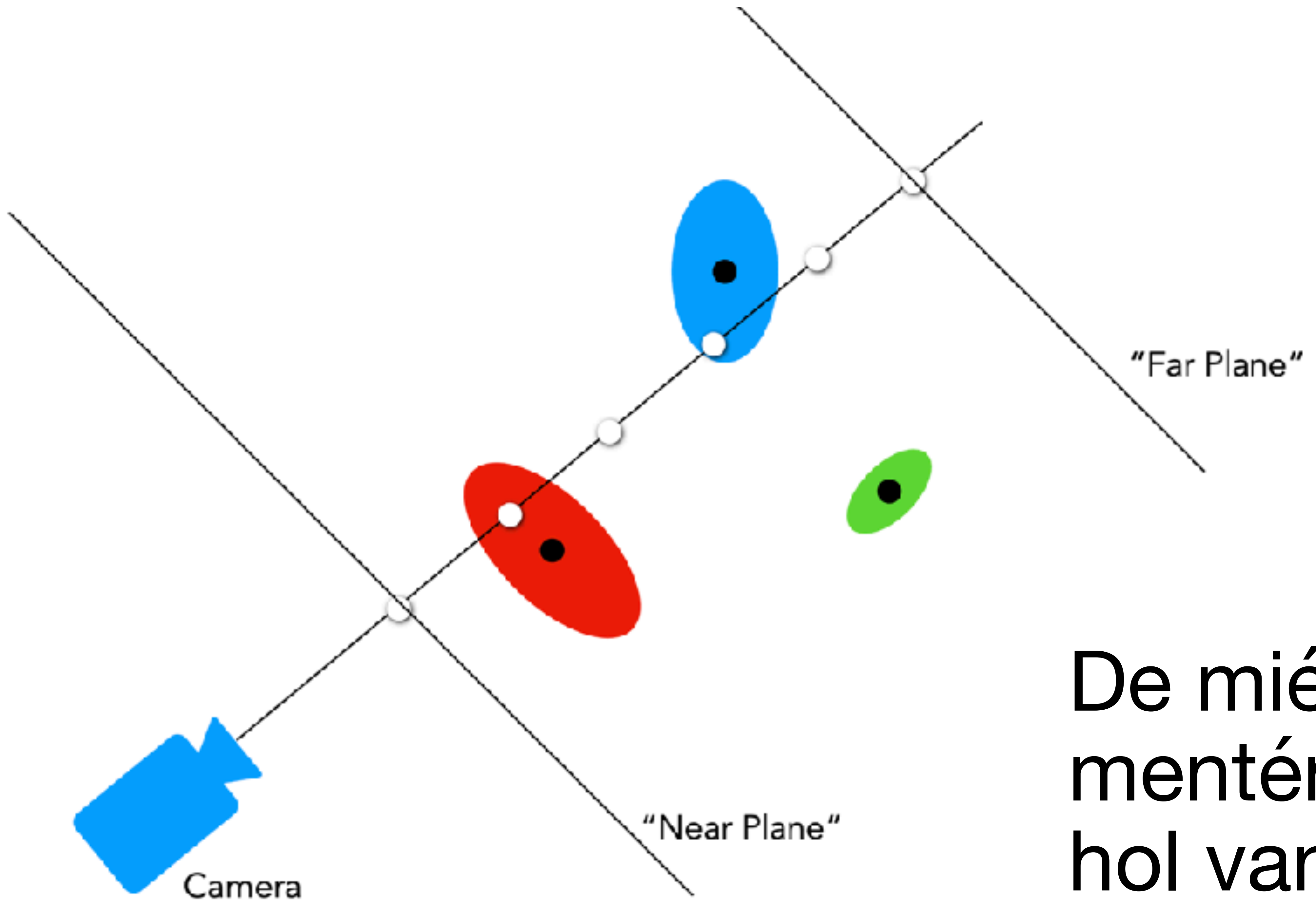


Slide credit: Vincent Sitzmann

3D Gaussian Splatting

Volume rendering?

Volume rendering?



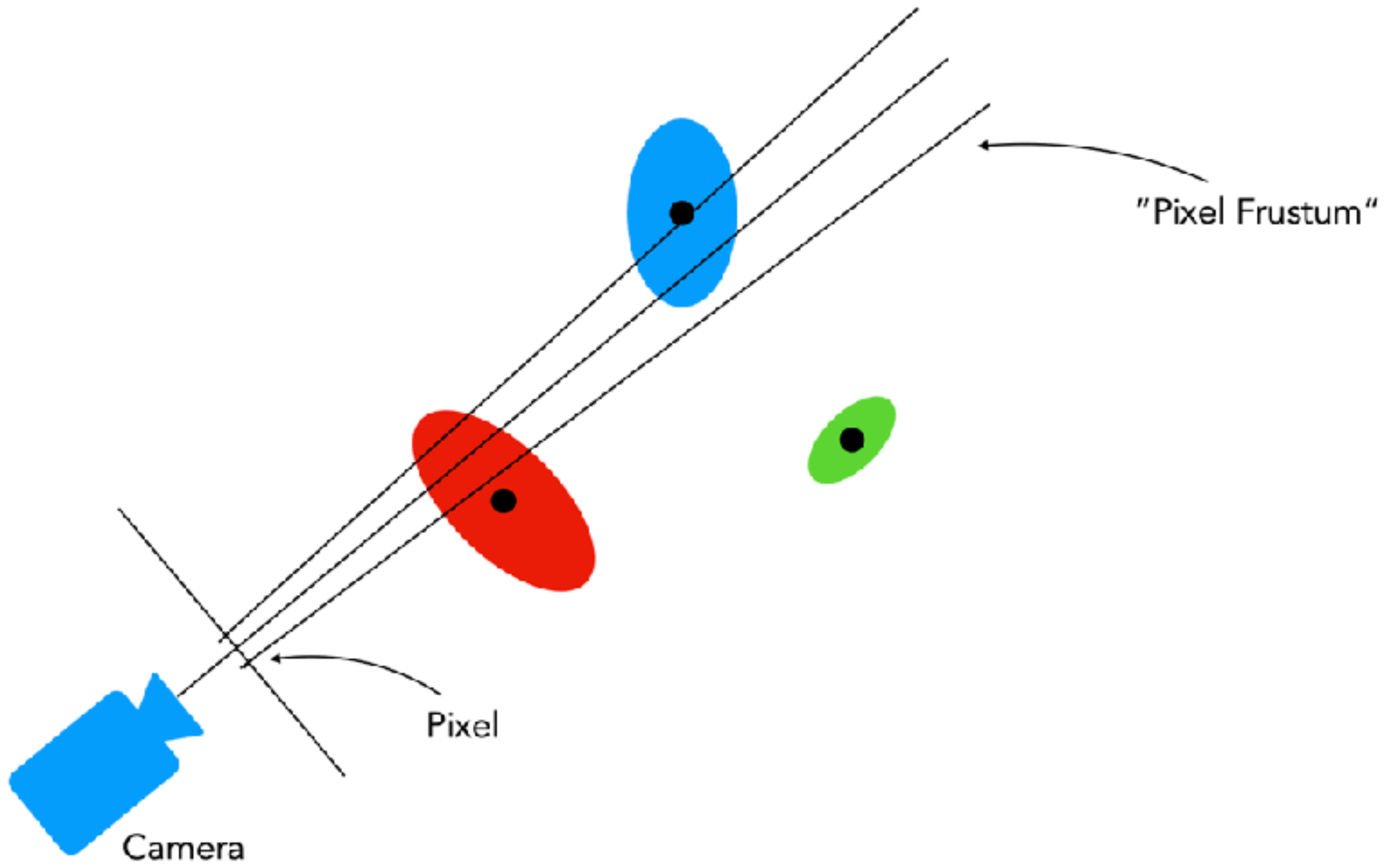
De miért lépkedjünk sugár mentén ha konkrétan tudjuk hol van "anyag"?

Slide credit: Vincent Sitzmann

3D Gaussian Splatting

Raszterizáció / Splatting

Volume rendering helyett: raszterizáció, “**splatting**”!

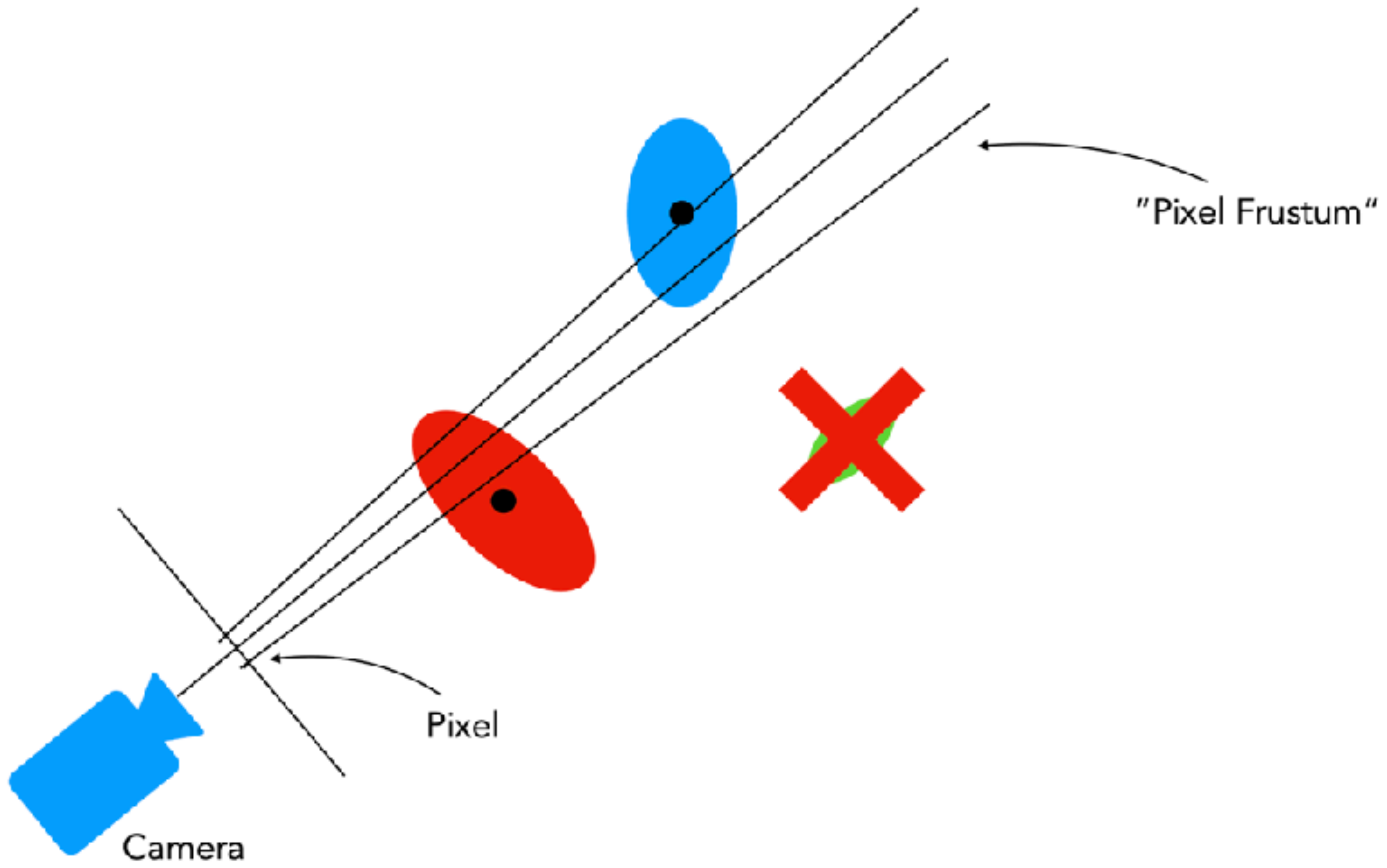


Slide credit: Vincent Sitzmann

3D Gaussian Splatting

Raszterizáció / Splatting

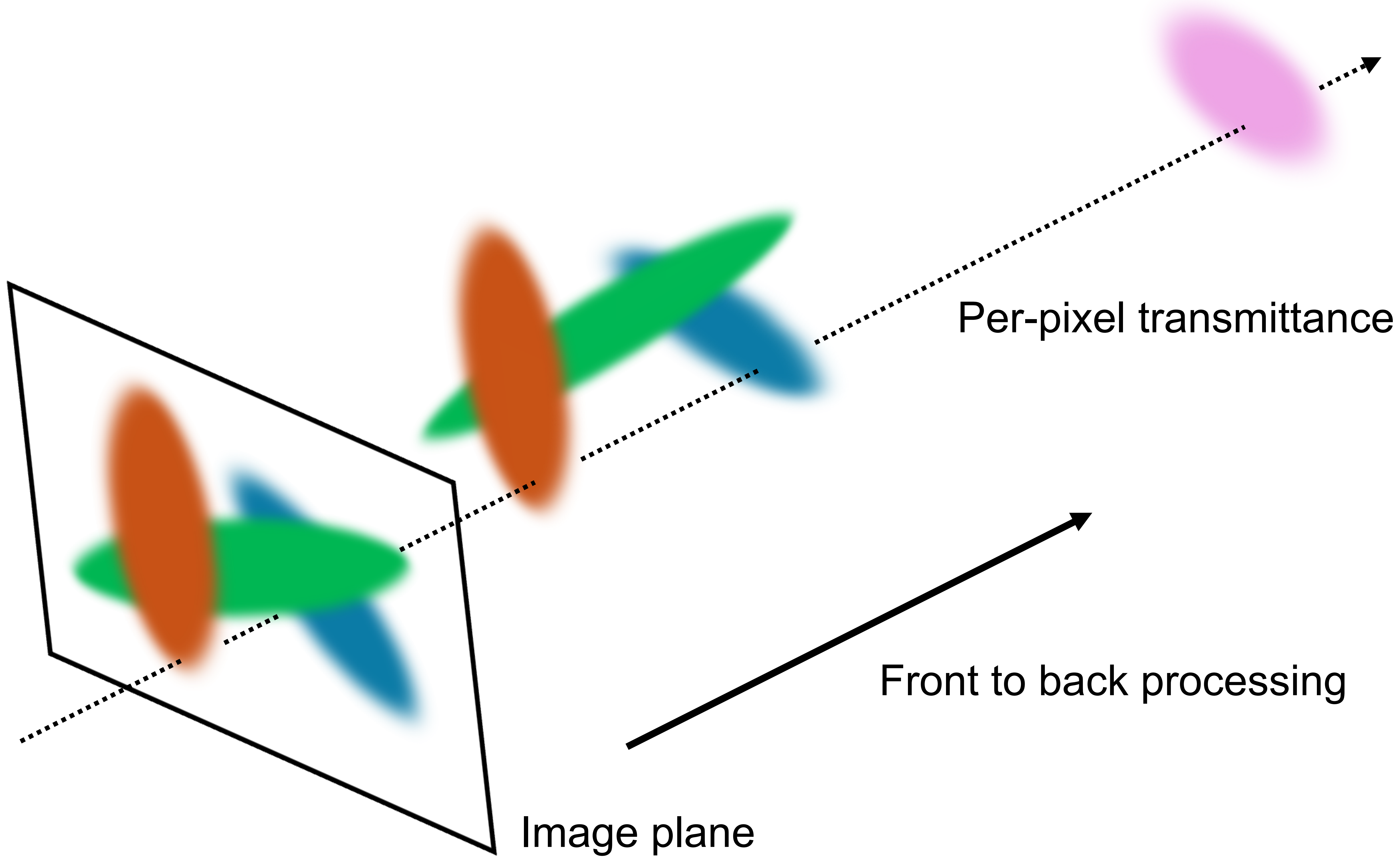
Ami frustrumon kívül esik, egyszerűen ignorálhatjuk…



Slide credit: Vincent Sitzmann

3D Gaussian Splatting

Raszterizáció / Splatting



3D Gaussian Splatting

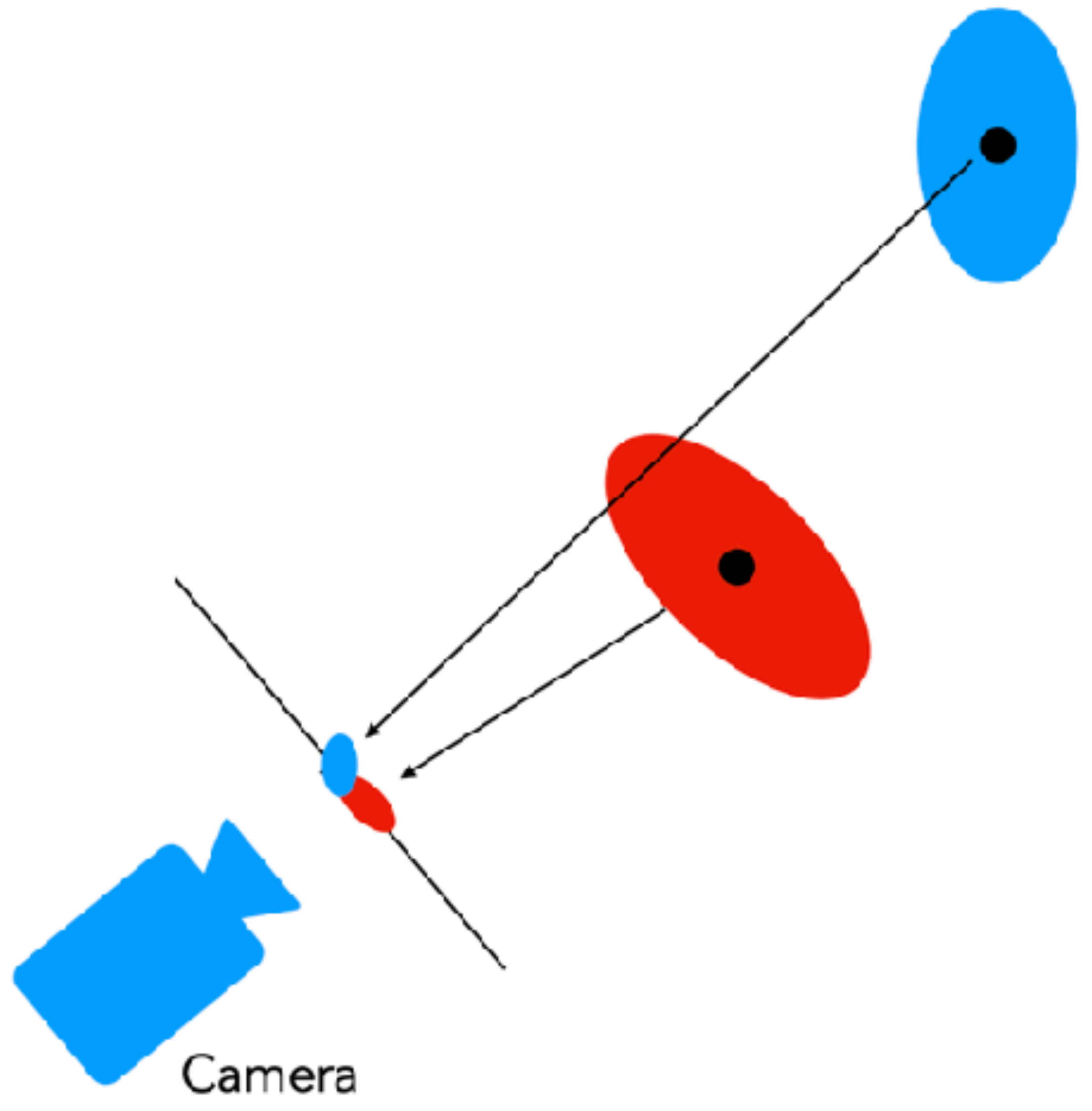
Raszterizáció / Splatting

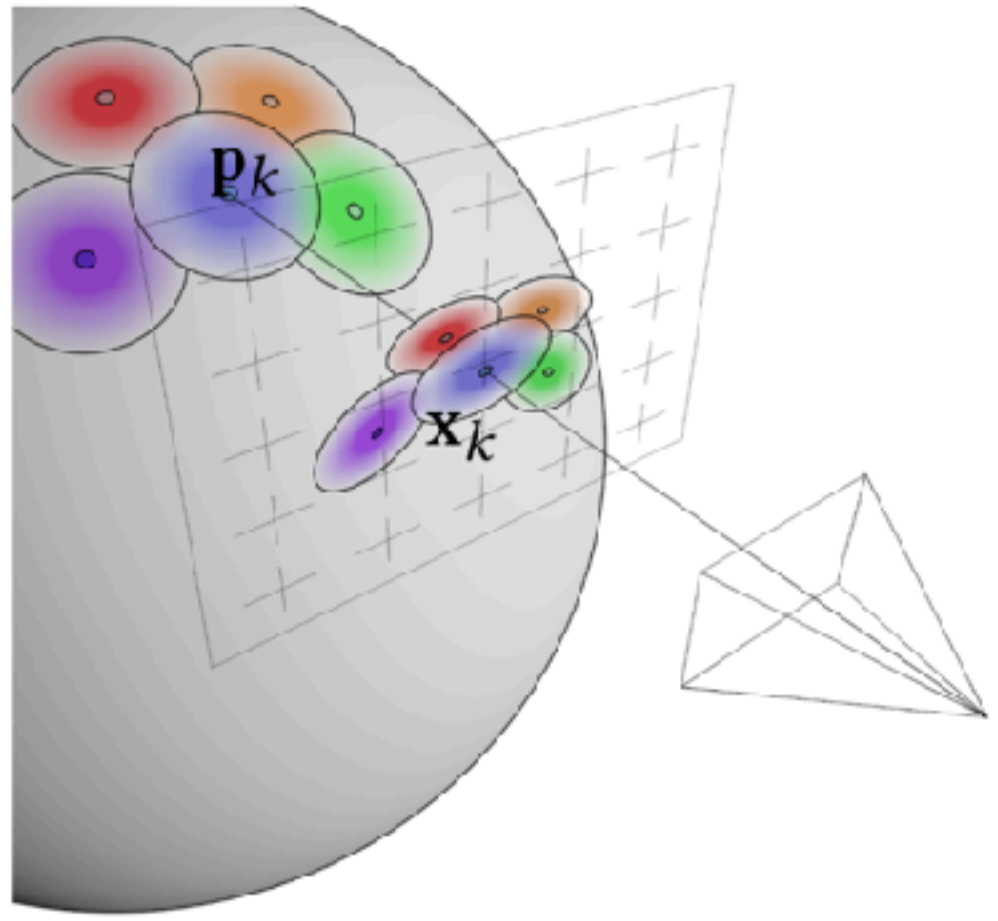
- Világ-Kamera transzformáció — affin leképezés: $\phi(x) = \mathbf{M}\mathbf{x} + \mathbf{p}$

- Gauss transzformációja szintén Gauss!

$$\mathcal{G}_{\mathbf{V}}(\mathbf{x}-\mathbf{p}) = \frac{1}{2\pi|\mathbf{V}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{p})^T \mathbf{V}^{-1}(\mathbf{x}-\mathbf{p})}$$

$$= \frac{1}{|\mathbf{M}^{-1}|} \mathcal{G}_{\mathbf{M}\mathbf{V}\mathbf{M}^T}(\mathbf{u}-\Phi(\mathbf{p}))$$





Slide credit: Vincent Sitzmann

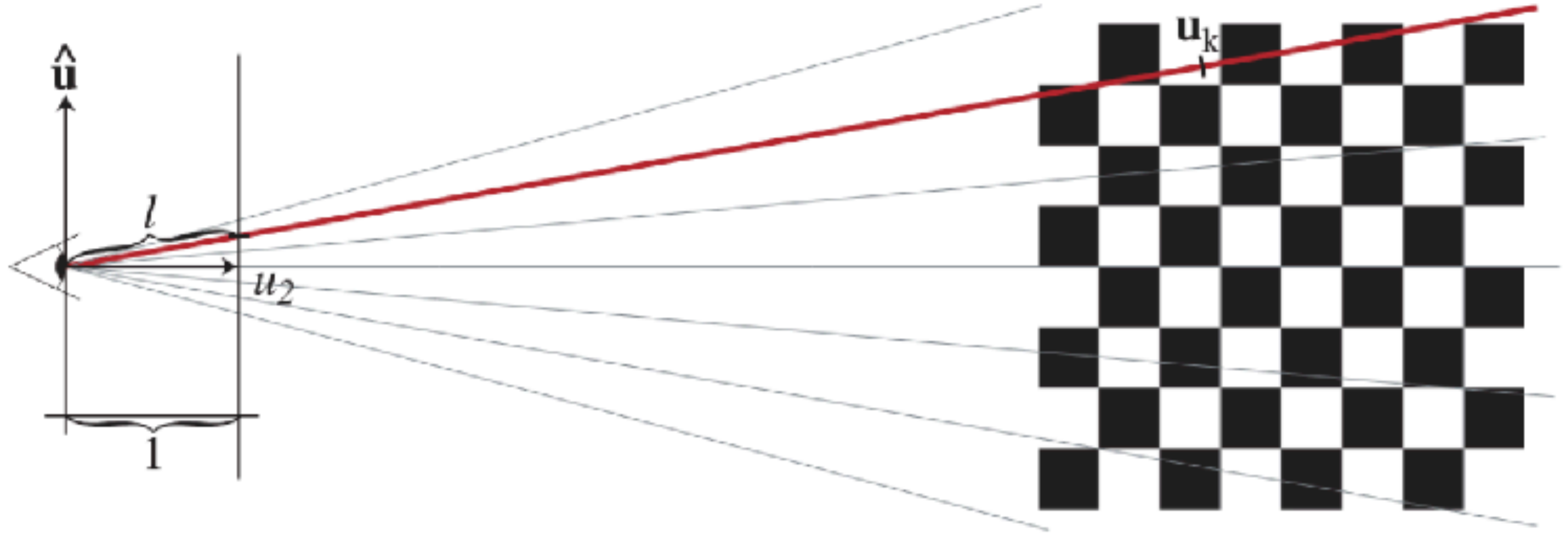
3D Gaussian Splatting

Raszterizáció / Splatting

- Perspektív projekció — nem affin transzformáció…

$$\begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \mathbf{m}(\mathbf{u}) = \begin{pmatrix} u_0/u_2 \\ u_1/u_2 \\ \|(u_0, u_1, u_2)^T\| \end{pmatrix}$$

$$\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} = \mathbf{m}^{-1}(\mathbf{x}) = \begin{pmatrix} x_0/l \cdot x_2 \\ x_1/l \cdot x_2 \\ 1/l \cdot x_2 \end{pmatrix},$$

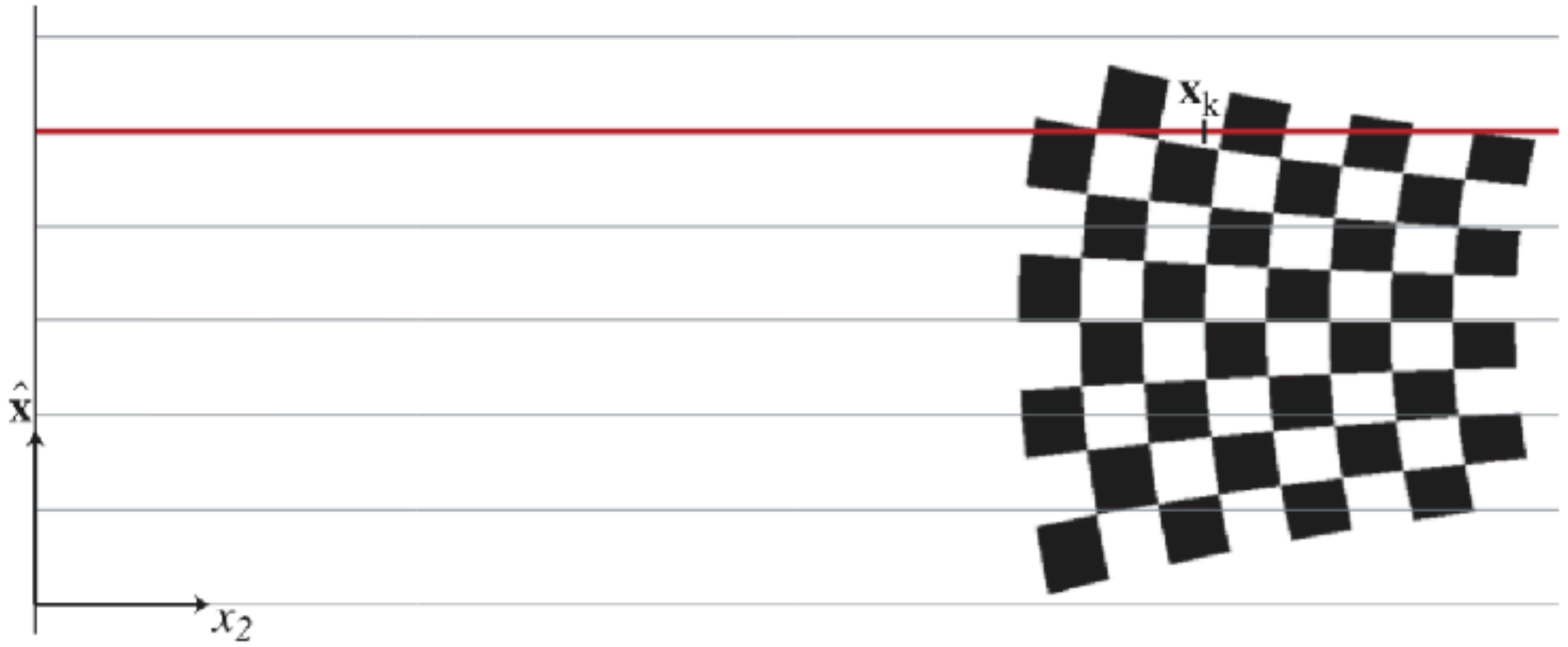


- Közelítés elsőfokú Taylor sorral (Jacobi mátrix):

$$\mathbf{m}_{\mathbf{u}_k}(\mathbf{u}) = \mathbf{x}_k + \mathbf{J}_{\mathbf{u}_k} \cdot (\mathbf{u} - \mathbf{u}_k) \qquad \mathbf{J}_{\mathbf{u}_k} = \frac{\partial \mathbf{m}}{\partial \mathbf{u}}(\mathbf{u}_k)$$

$$\frac{1}{|\mathbf{W}^{-1}||\mathbf{J}^{-1}|} \mathcal{G}_{\mathbf{V}_k}(\mathbf{x} - \mathbf{x}_k)$$

$$\begin{aligned} \mathbf{V}_k &= \mathbf{J}\mathbf{V}_k'\mathbf{J}^T \\ &= \mathbf{J}\mathbf{W}\mathbf{V}_k''\mathbf{W}^T\mathbf{J}^T \end{aligned}$$



3D Gaussian Splatting

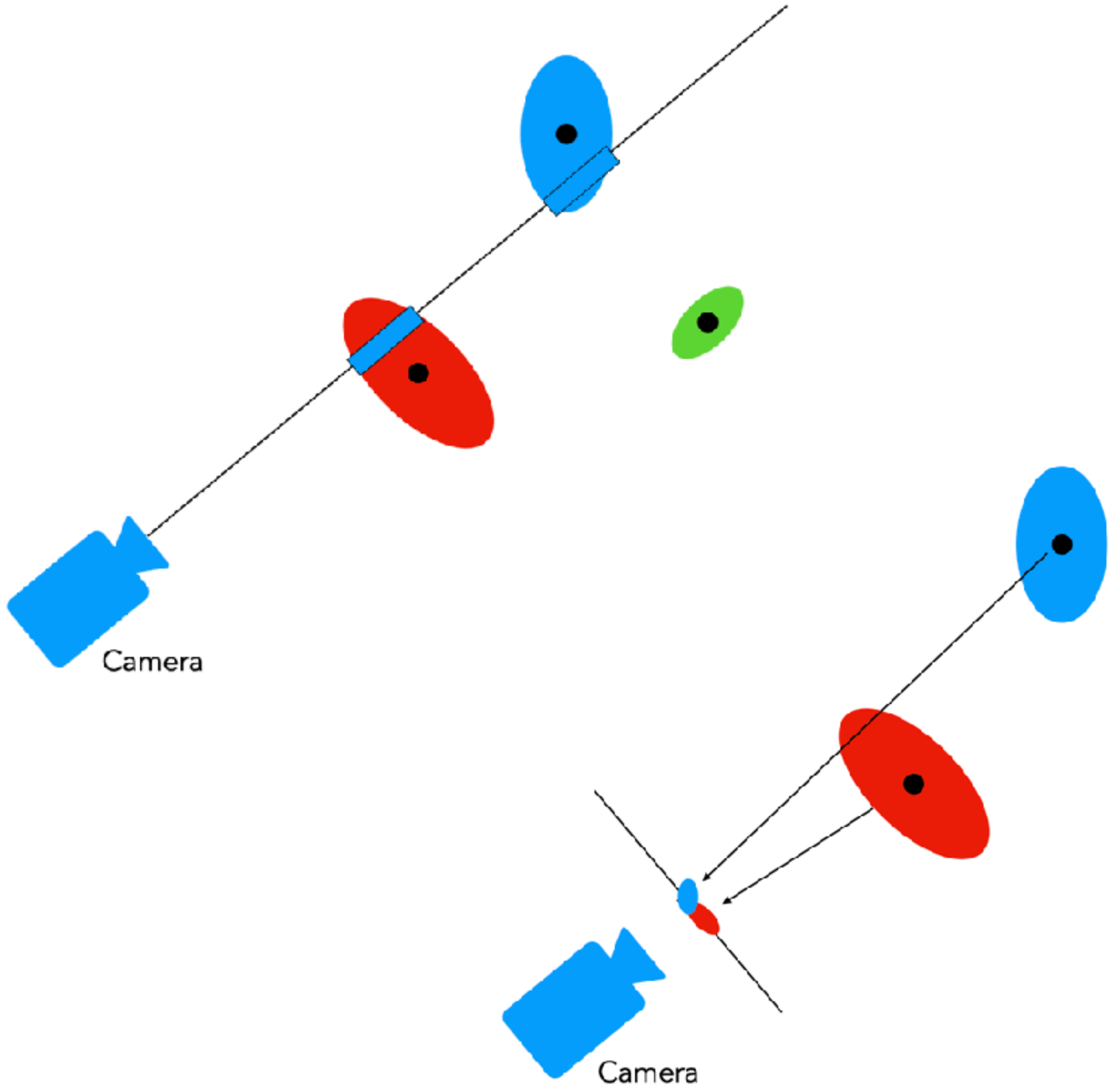
Raszterizáció / Splatting

- 3D Gauss sugár menti integrálja = 2D Gauss

$$\int_{\mathbb{R}} \mathcal{G}^3_{\mathbf{V}}(\mathbf{x}-\mathbf{p})\,dx_2 = \mathcal{G}^2_{\hat{\mathbf{V}}}(\hat{\mathbf{x}}-\hat{\mathbf{p}})$$

$$\mathbf{V} = \begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix} \Leftrightarrow \begin{pmatrix} a & b \\ b & d \end{pmatrix} = \hat{\mathbf{V}}$$

$$\begin{aligned} q_k(\hat{\mathbf{x}}) &= \int_{\mathbb{R}} \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\mathbf{V}_k}(\hat{\mathbf{x}}-\hat{\mathbf{x}}_k, x_2 - x_{k2})\,dx_2 \\ &= \frac{1}{|\mathbf{J}^{-1}||\mathbf{W}^{-1}|} \mathcal{G}_{\hat{\mathbf{V}}_k}(\hat{\mathbf{x}}-\hat{\mathbf{x}}_k) \end{aligned}$$



3D Gaussian Splatting

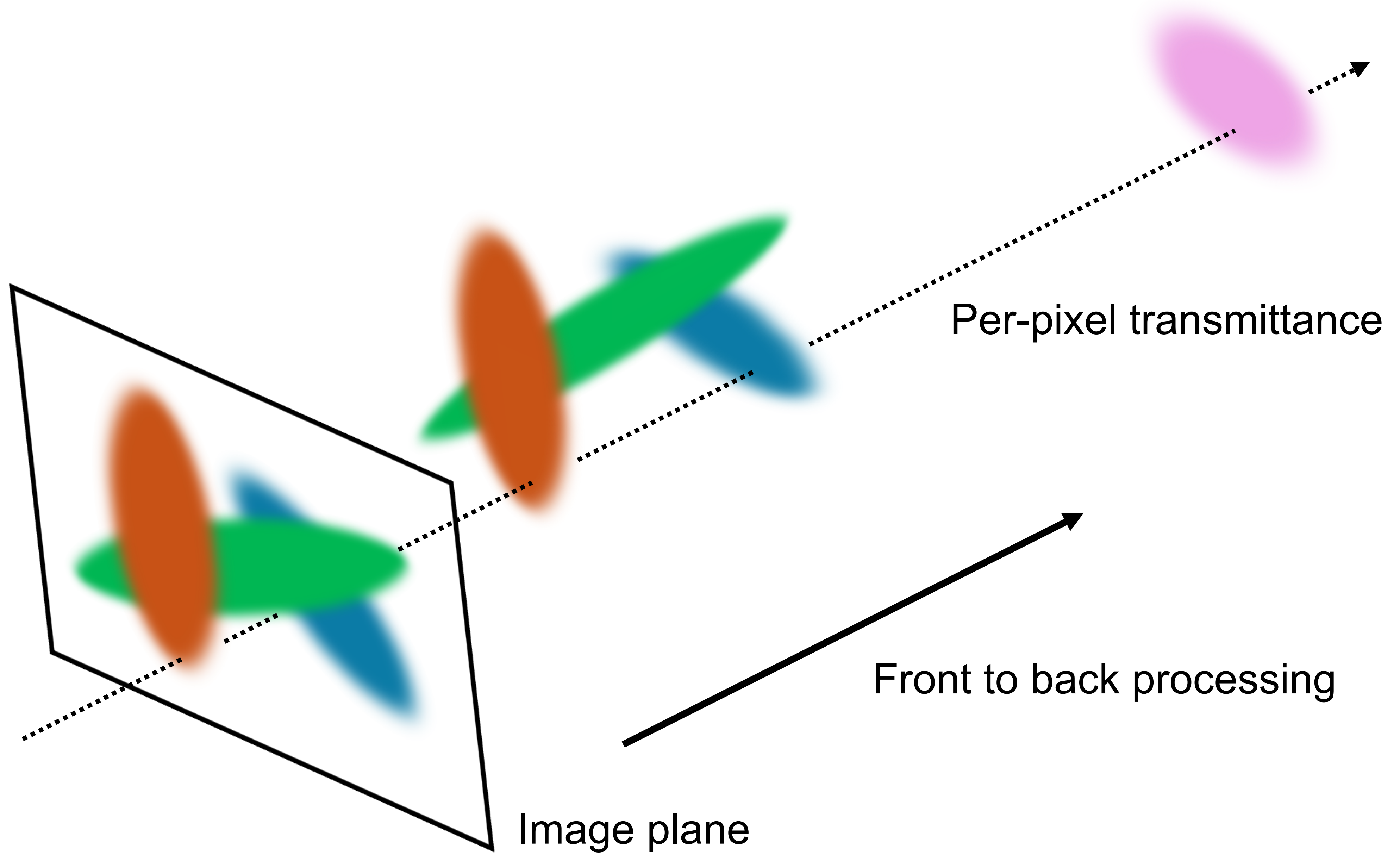
Raszterizáció / Splatting

- Végeredmény: alpha blending!

$$C_i = \sum_{n \le N} c_n \cdot \alpha_n \cdot T_n, \quad \text{where} \quad T_n = \prod_{m<n} (1 - \alpha_m)$$

$$\alpha_n = o_n \cdot \exp(-\sigma_n), \quad \sigma_n = \frac{1}{2}\Delta_n^\top \Sigma'^{-1} \Delta_n$$

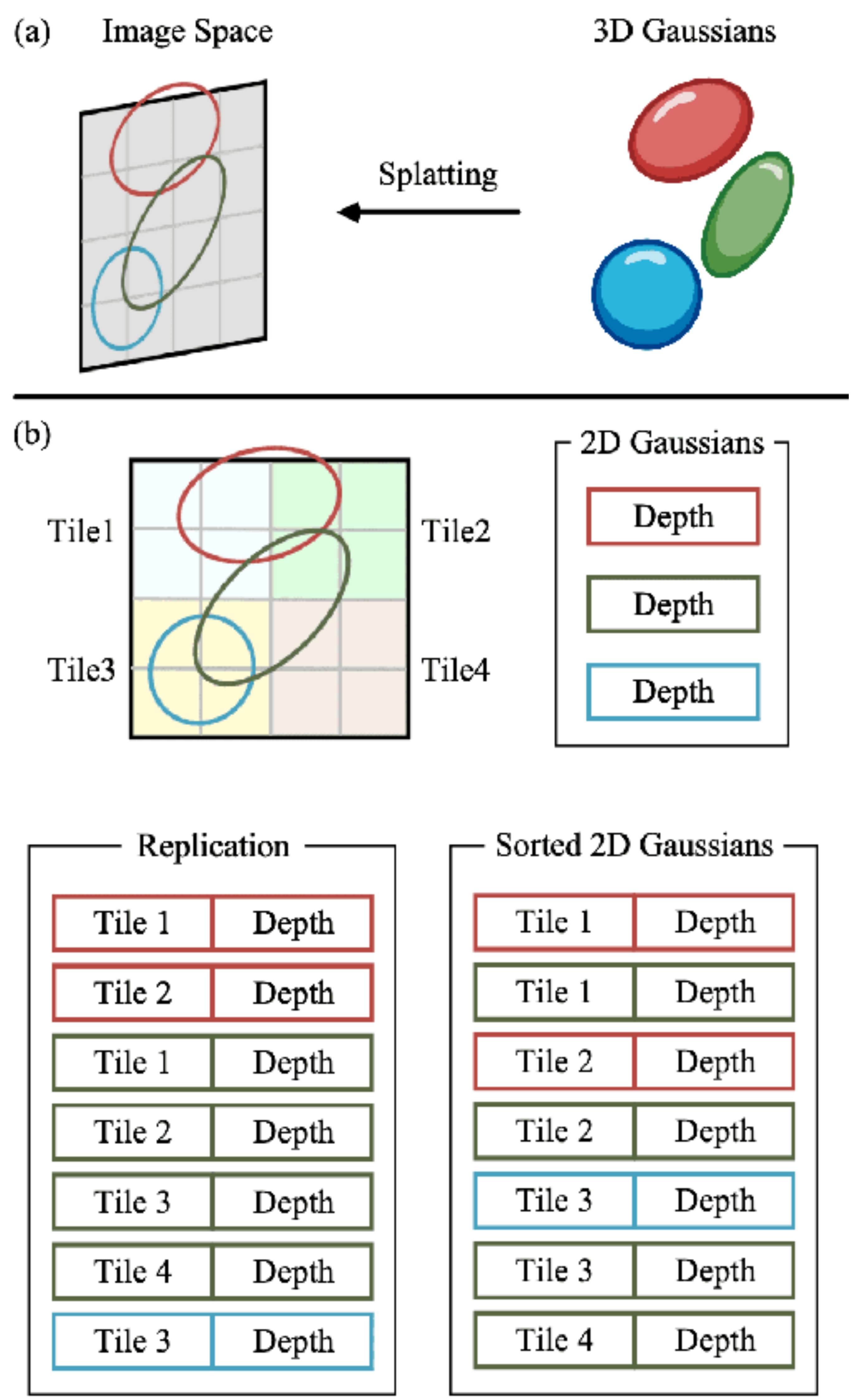
$$\Delta_n = \mu_n - p_i$$



3D Gaussian Splatting

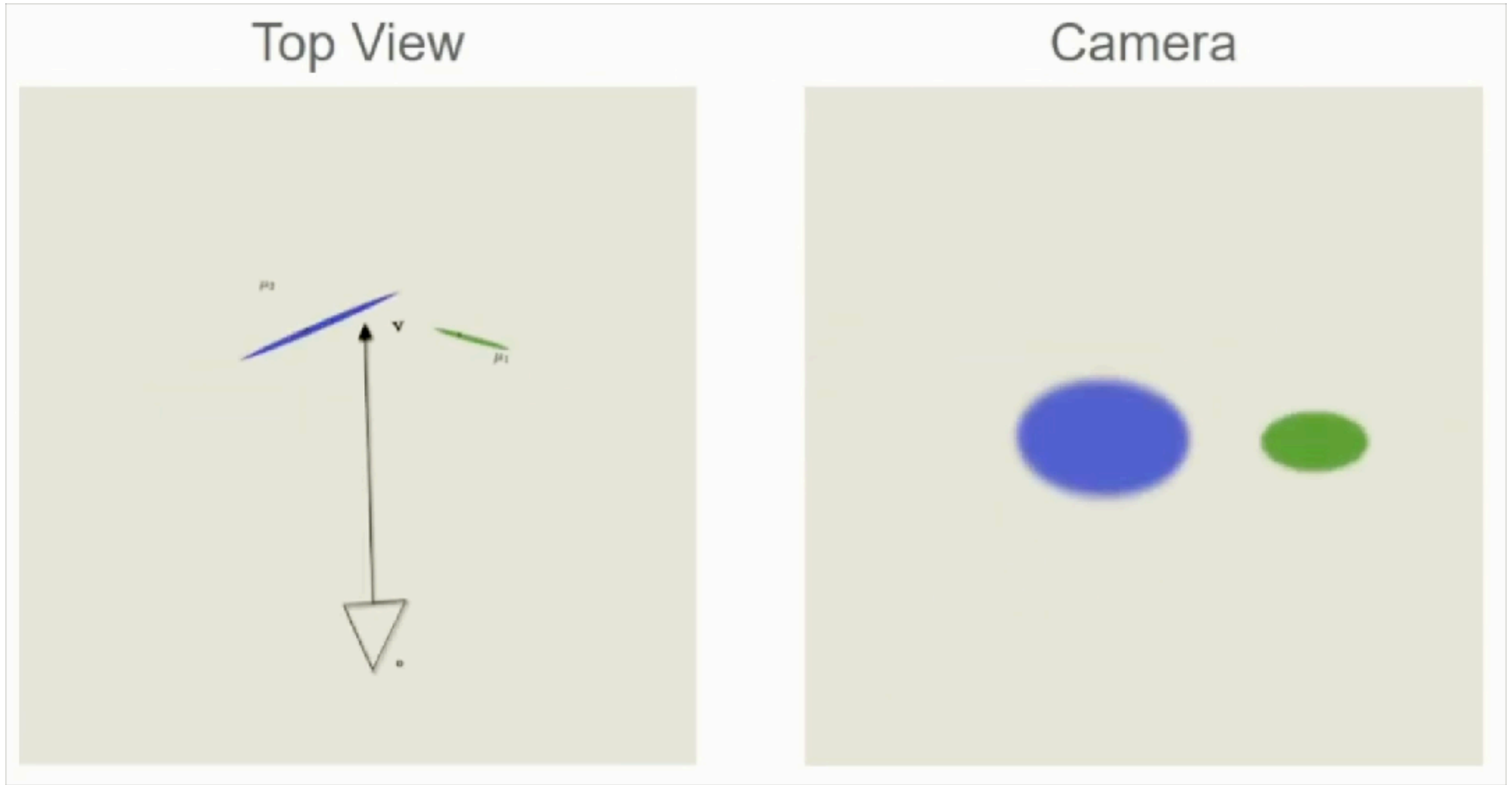
Raszterizáció / Splatting

- Alpha blendinghez kamerához képest **rendezni** szükséges a Gauss-okat…
- Per-pixel rendezés: túl költséges…
- *Globális* rendezés mélység szerint — párhuzamosított **GPU radix sort**
- Raszterezés — GPU implementáció: **tile-alapú** (16x16 pixel) párhozamosított rendering
  - Gauss-ok replikációja átfedő tile-okban
  - Rendezés tile index & mélység szerint!



3D Gaussian Splatting

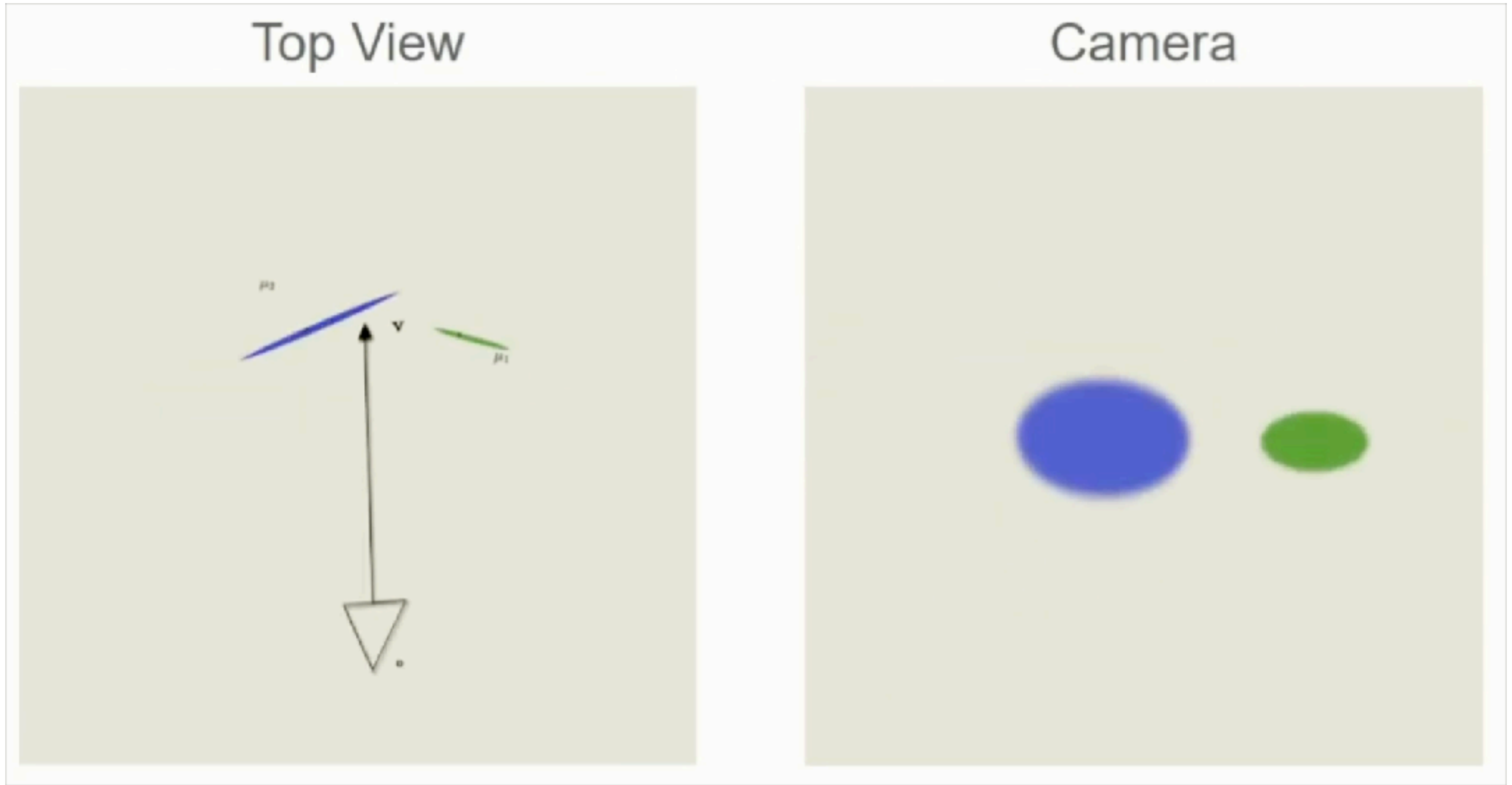
Raszterizáció / Splatting



Rendezés a középpontok mélysége alapján:
nem teljesen korrekt (**“popping artifact**”), kompromisszumos megoldás…

3D Gaussian Splatting

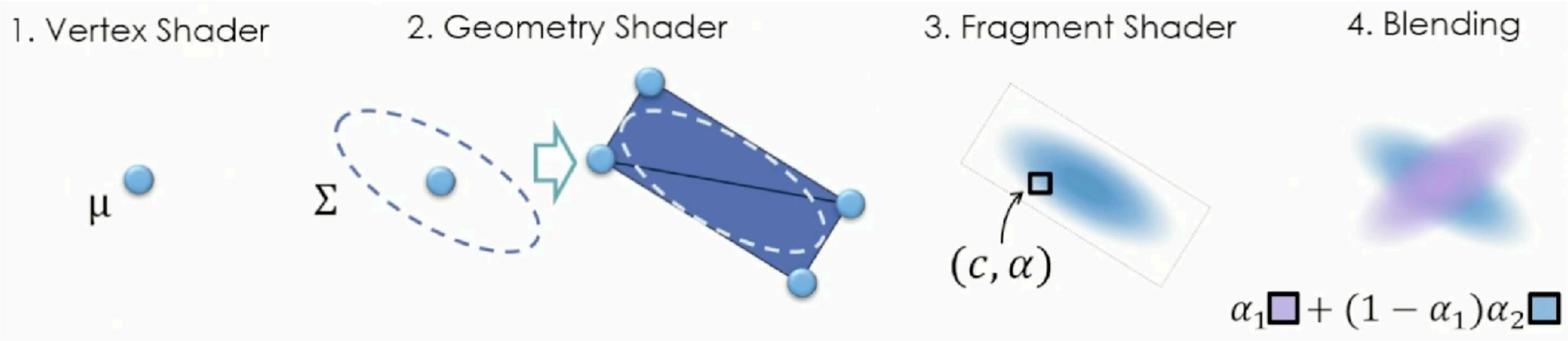
Raszterizáció / Splatting



Rendezés a középpontok mélysége alapján:
nem teljesen korrekt (**“popping artifact”**), kompromisszumos megoldás…

3D Gaussian Splatting

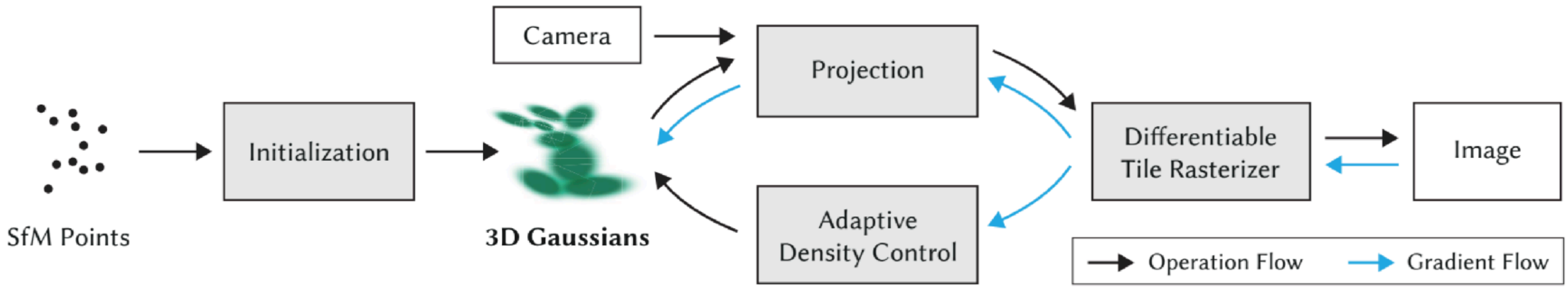
Raszterizáció / Splatting



3DGS forward rendering: <u>standard grafikai csővezetékkel implementálható!</u>

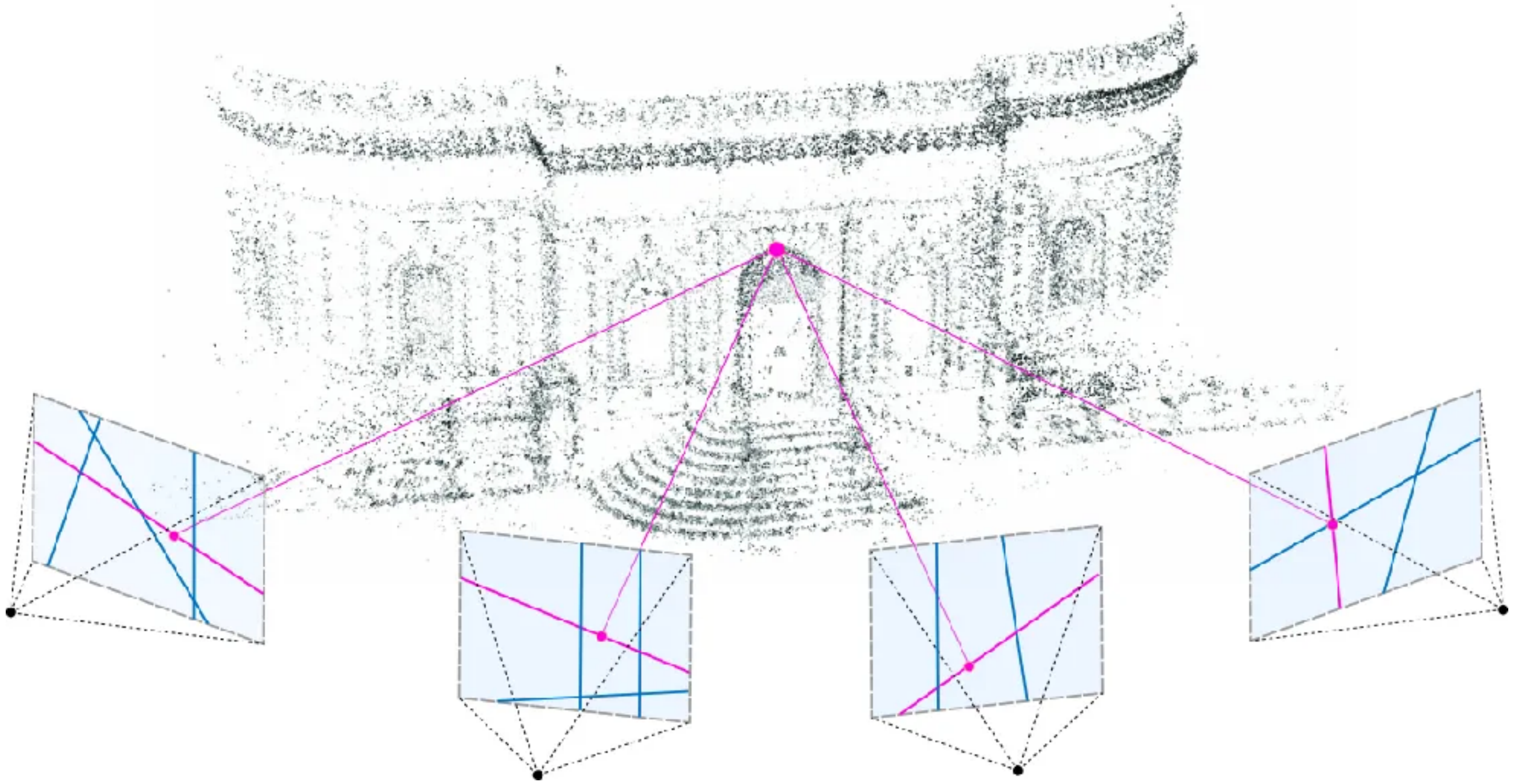
3D Gaussian Splatting

Optimalizáció — Pipeline



3D Gaussian Splatting

Optimalizáció — Inicializáció



Lehet random pontfelhővel is inicializálni…

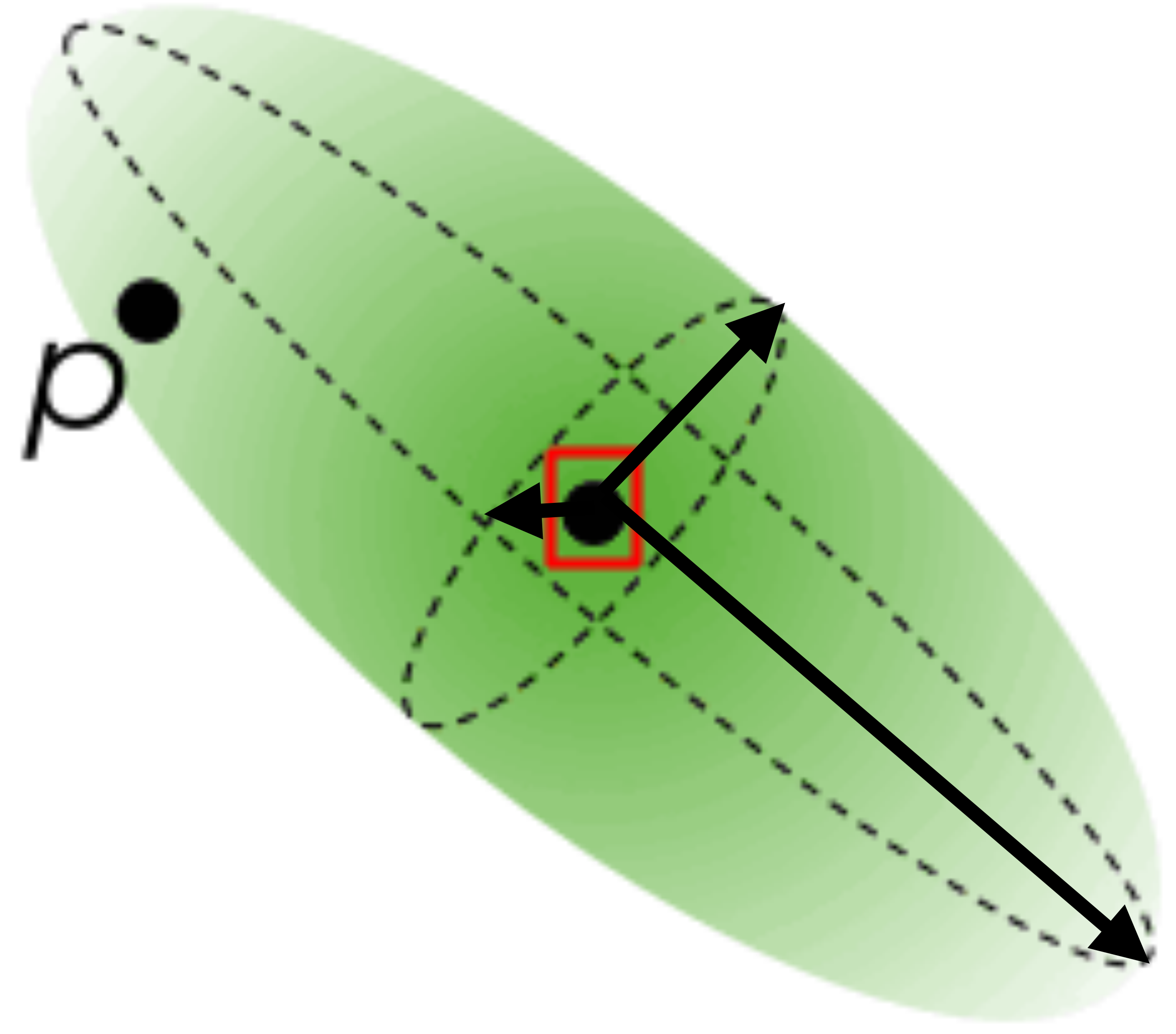
Inicializálás: klasszikus fotogrammetria, **Structure-from-Motion (SfM)** (COLMAP):
Kamera pozíciók + Ritka pontfelhő

3D Gaussian Splatting

Optimalizáció — Paraméterezés

- Hogy garantáljuk, hogy a kovariancia mátrix poz. def. marad optimalizáció során?
- Faktorizáció: forgató mátrix (kvaternió) & skálázás

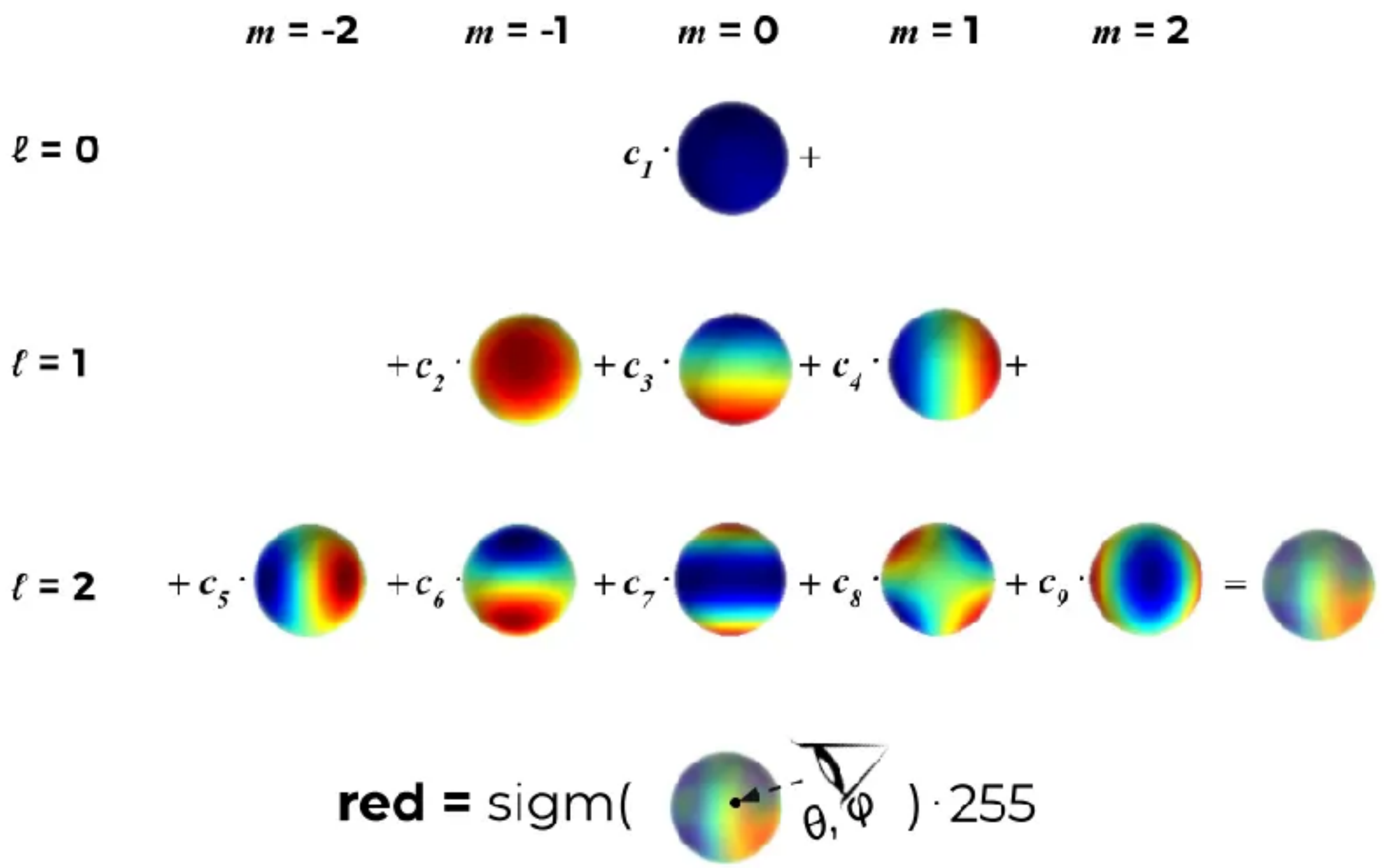
$$\Sigma = RSS^T R^T \qquad S = \mathrm{diag}(s_x, s_y, s_z)$$



- σ: szigmoid aktiváció
- S: exponenciális aktiváció

3D Gaussian Splatting

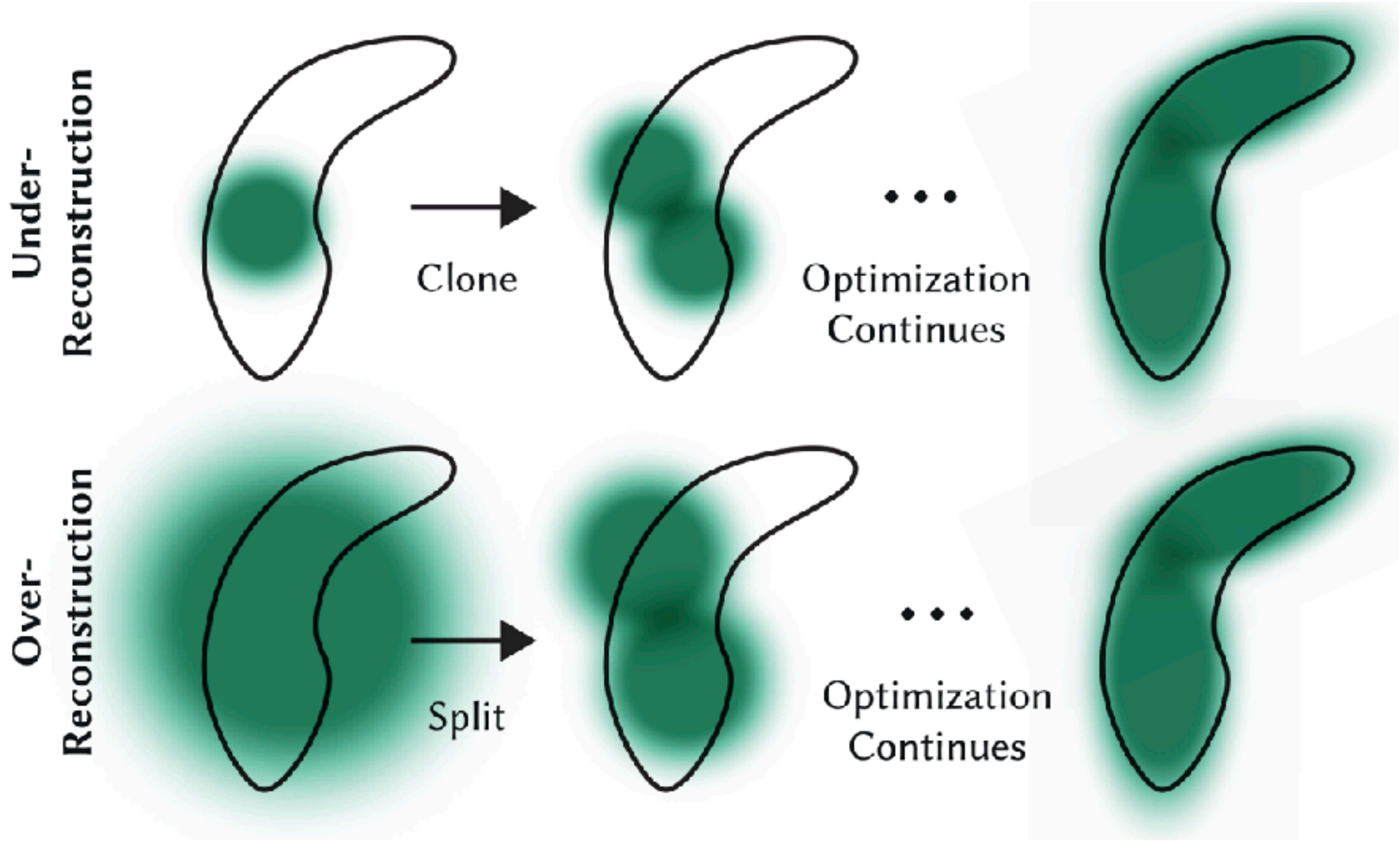
Optimalizáció — Paraméterezés



Nézeti irány függő szín — **<u>gömbi harmonikusok</u>**

3D Gaussian Splatting
Optimalizáció — Implementáció

- Optimalizáció: **Adam** algoritmus
  - Explicit gradiens formulák a hatékonyság érdekében!
- Periodikus **Gauss sűrítés/ritkítás**:
  - Nagy pozíció-gradiensek esetén Gauss-ok duplikálása
  - Felesleges (pl. áttetsző) Gauss-ok eldobása
- Gömbi harmonikus “sávok” fokozatos bekapcsolása
- 30k Adam iteráció — kb. 15-30 perc alatt konvergál (kommersz NVIDIA GPU-n)



3D Gaussian Splatting
Implementációk

https://github.com/graphdeco-inria/gaussian-splatting

Efficient
Gaussian Splatting
CUDA Backend

https://docs.gsplat.studio/

https://docs.nerf.studio/nerfology/methods/splat.html

A high-performance C++ and CUDA implementation of 3D Gaussian Splatting

https://lichtfeld.io/

3D Gaussian Splatting

Implementációk

https://github.com/graphdeco-inria/gaussian-splatting

Efficient Gaussian Splatting CUDA Backend

https://docs.gsplat.studio/

https://docs.nerf.studio/nerfology/methods/splat.html

A high-performance C++ and CUDA implementation of 3D Gaussian Splatting

https://lichtfeld.io/

3D Gaussian Splatting

Implementációk — Web-alapú renderek

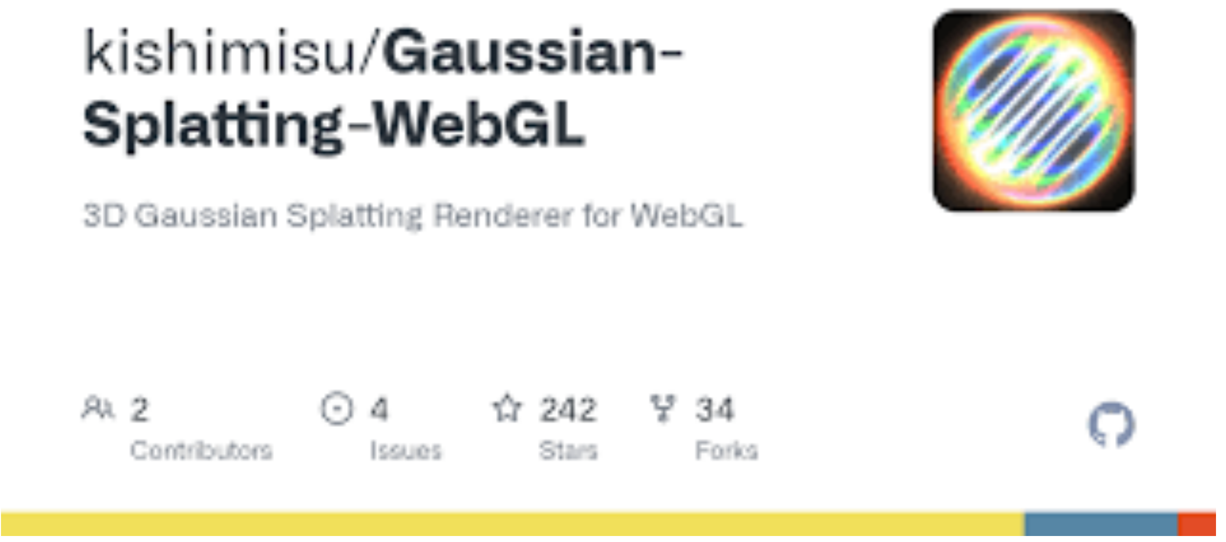
SuperSplat

https://superspl.at/

SPARK

An advanced 3D Gaussian Splatting renderer for THREE.js

https://sparkjs.dev/



https://webgl-gaussian-splatting.vercel.app/

Luma WebGL Library

https://lumalabs.ai/luma-web-library

3D Gaussian Splatting
Kommerciális implementációk

Polycam

https://poly.cam/tools/gaussian-splatting

postshot

https://www.jawset.com/

NIANTIC SCANIVERSE

https://scaniverse.com/

KIRI Engine

https://www.kiriengine.app/

LUMA AI

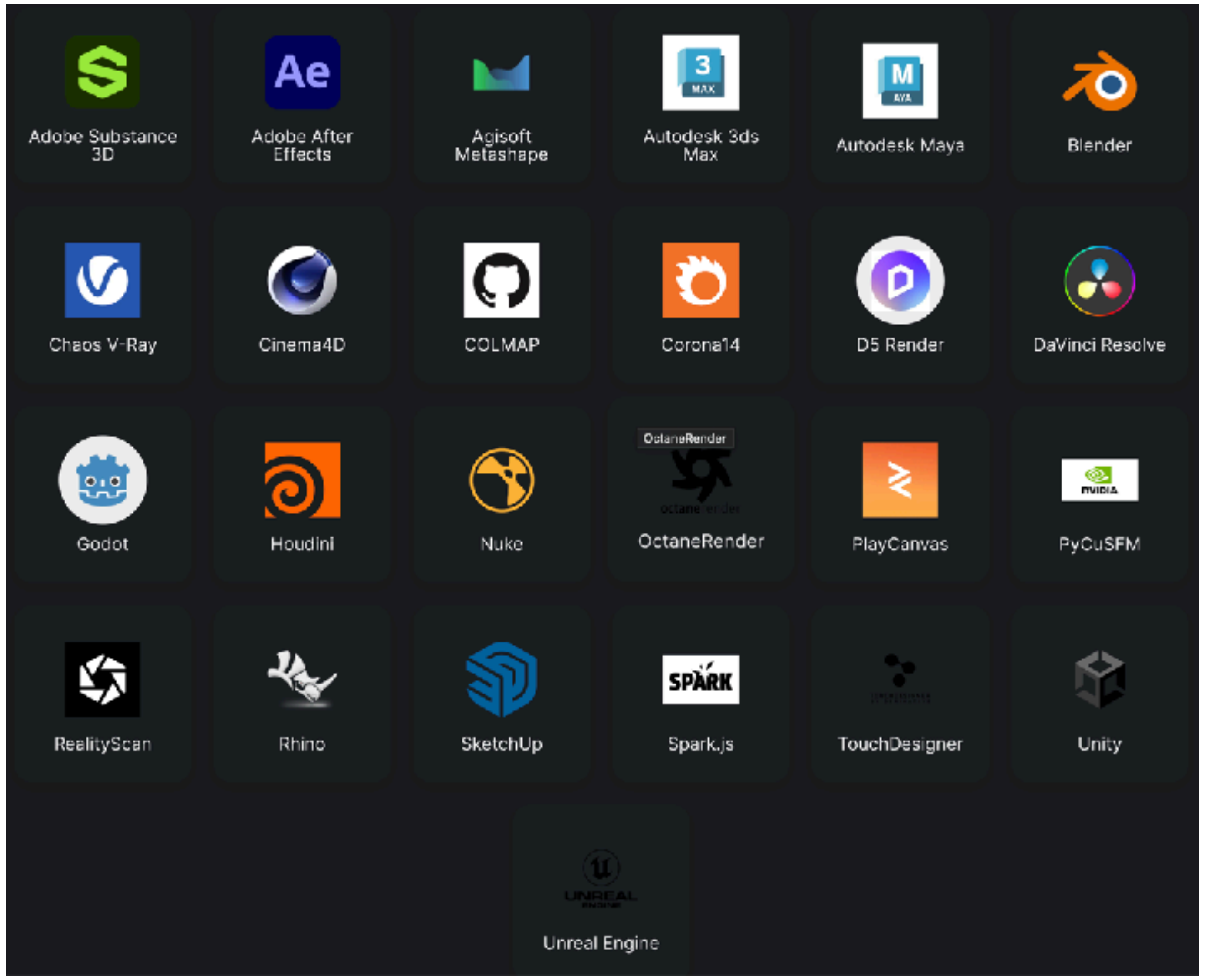
https://lumalabs.ai/featured

SpatialView

https://spatialview.io/

3D Gaussian Splatting

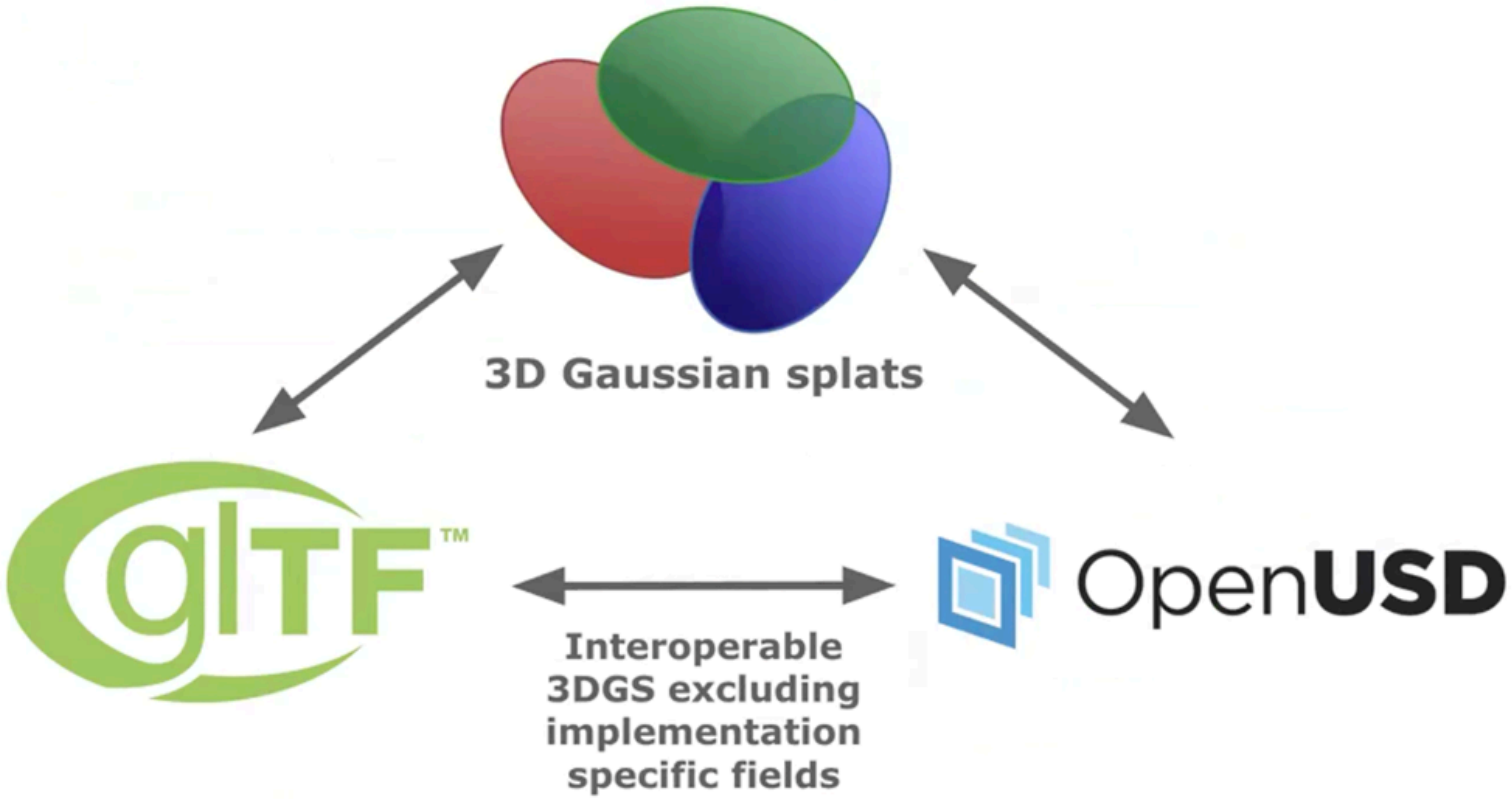
Ipari adoptáció



https://radiancefields.com/3d-gaussian-splatting-engine-support

3D Gaussian Splatting

Szabványosítás



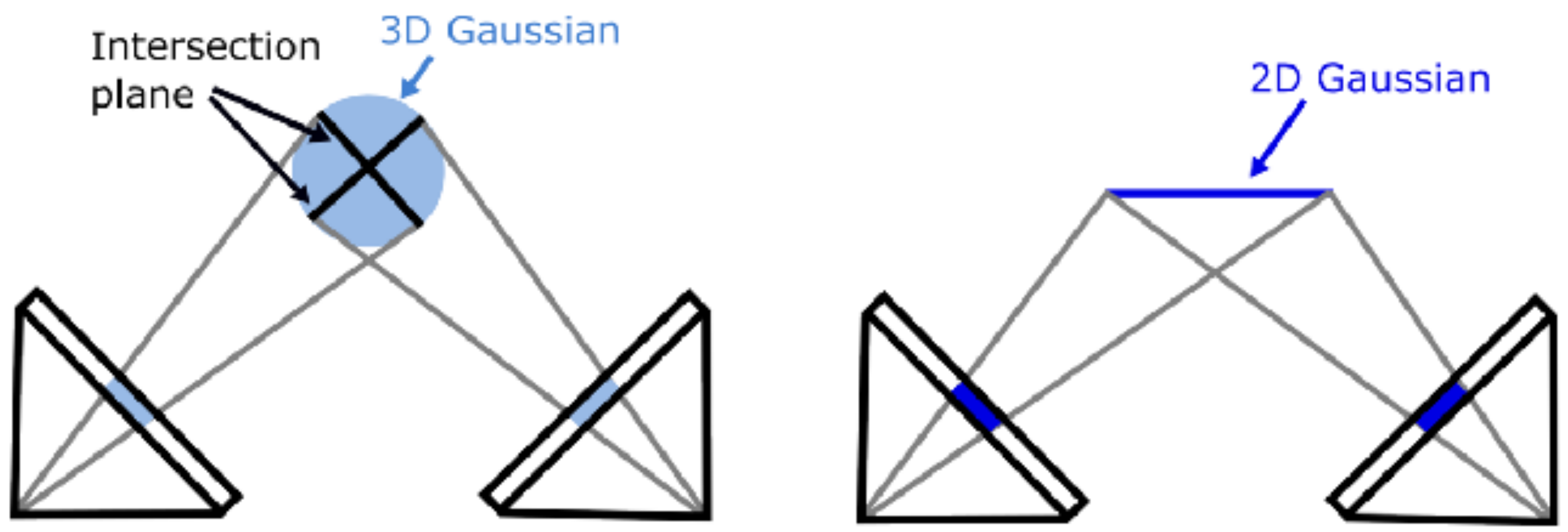
https://www.khronos.org/blog/khronos-ogc-and-geospatial-leaders-add-3d-gaussian-splats-to-the-gltf-asset-standard

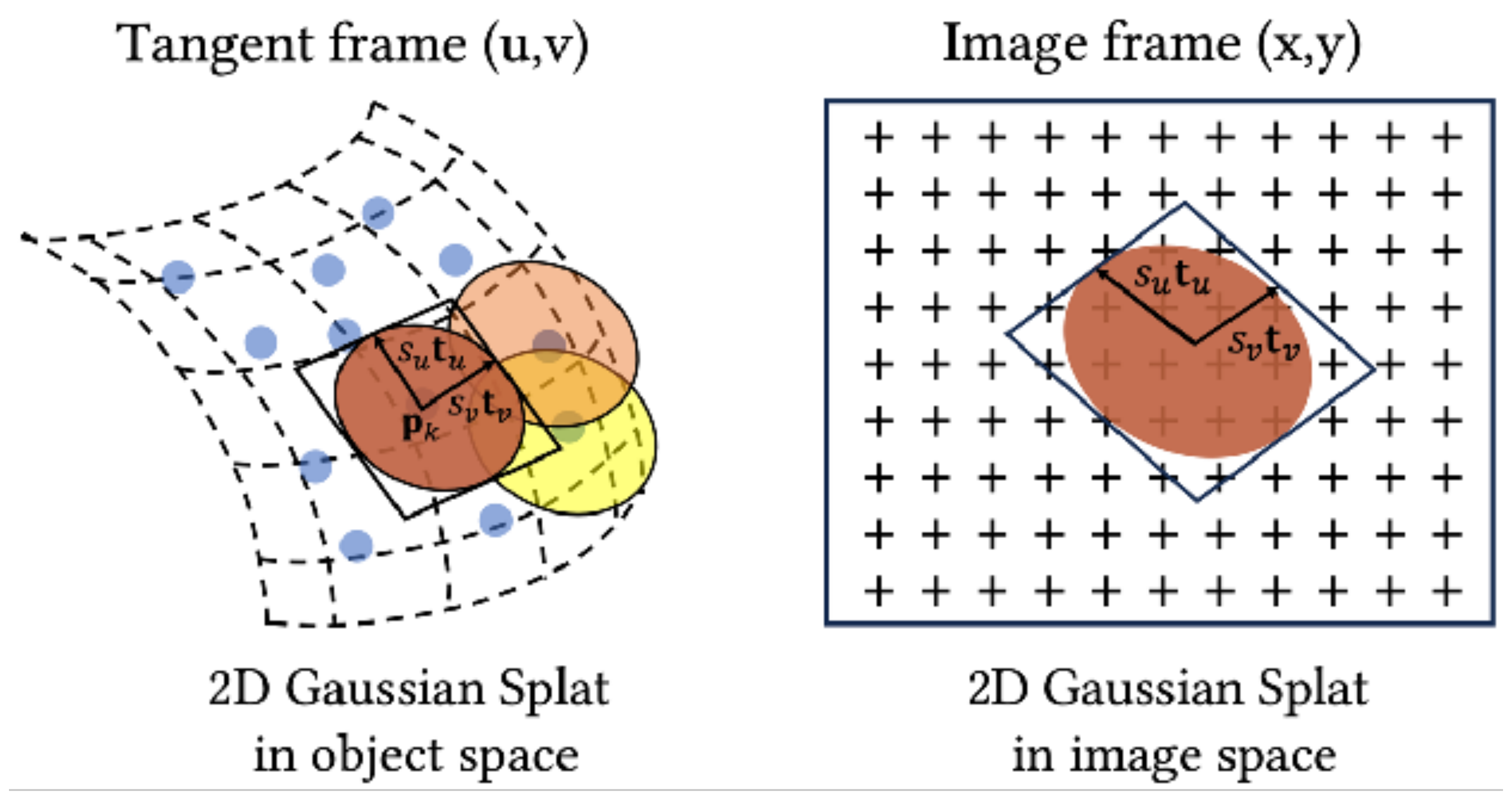
3D Gaussian Splatting

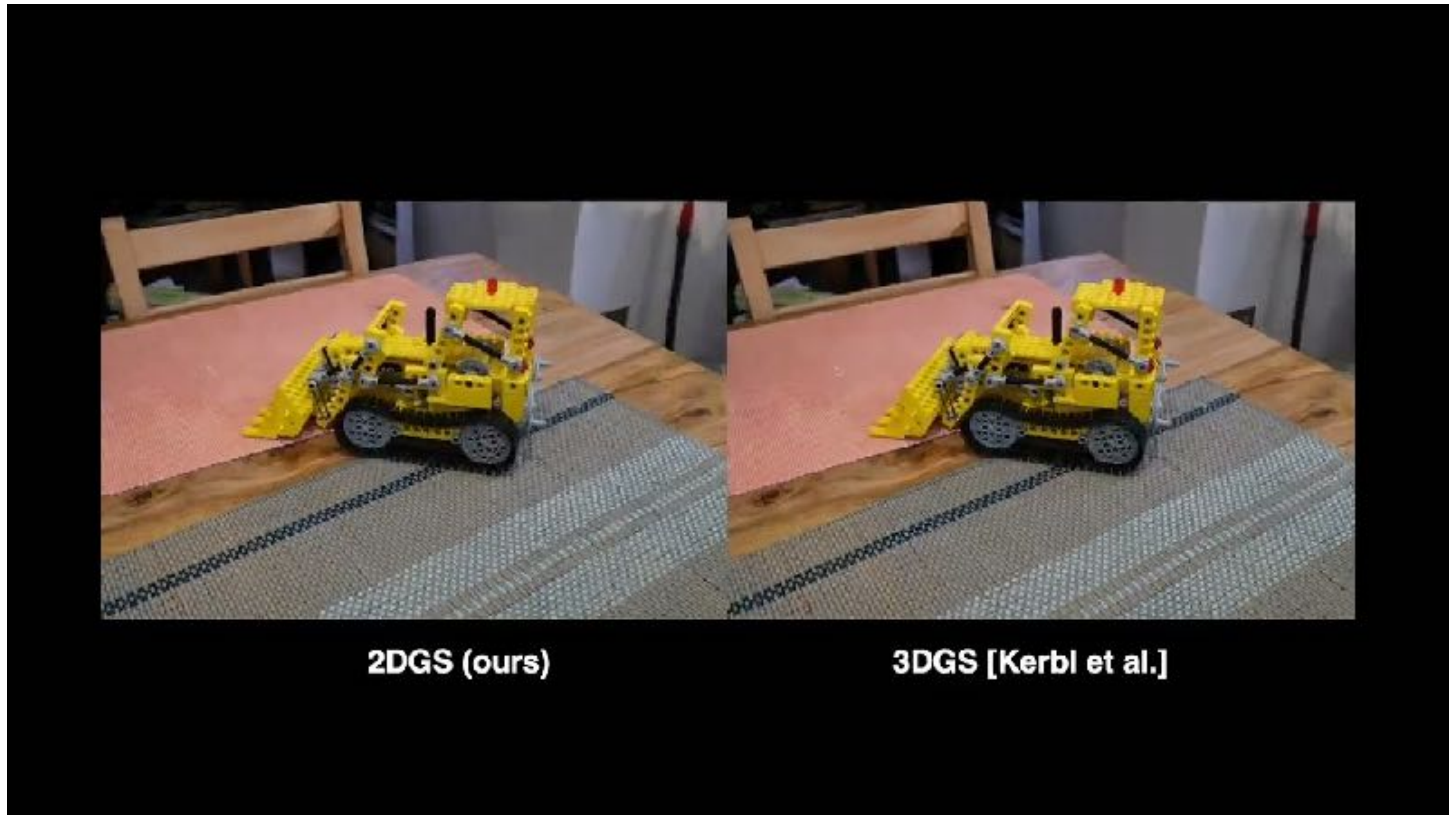
Limitációk

- “Vanilla” 3DGS limitációi:
  - Felületek nem élesek
  - SfM inicializáció (néha lassabb mint a 3DGS optimalizáció!)
  - Pinhole kameramodell
  - Közelítő renderelés
  - Nagy memóriaigény
  - Statikus jelenetek

3D Gaussian Splatting
Variánsok — 2DGS

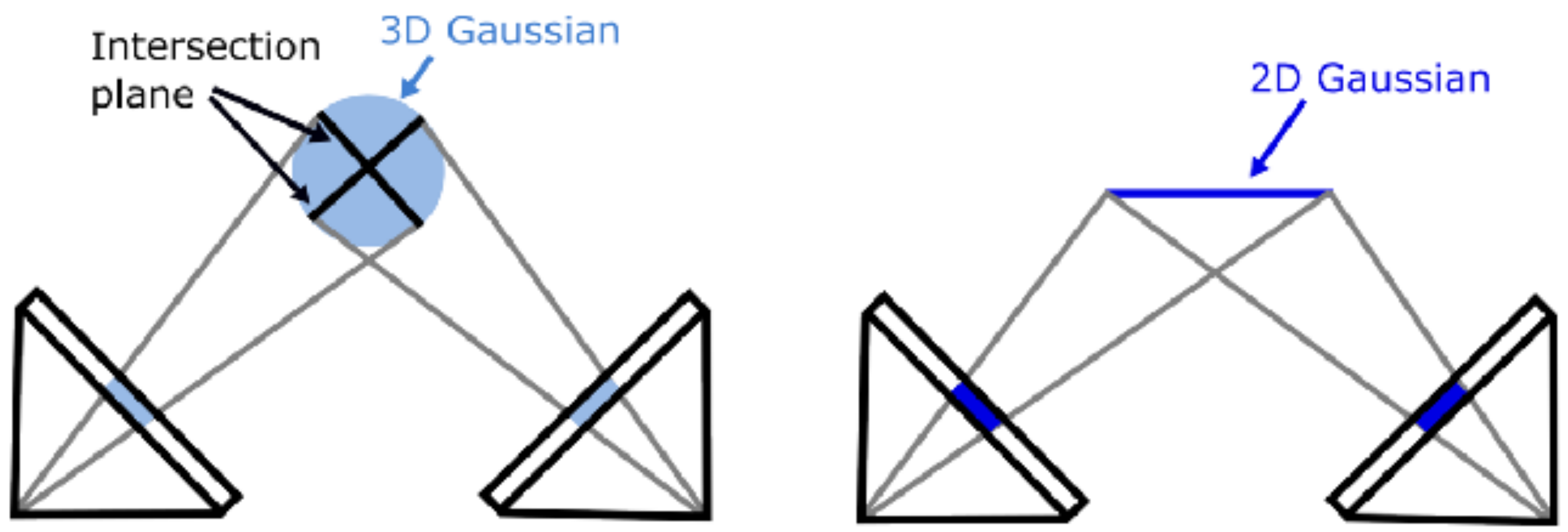


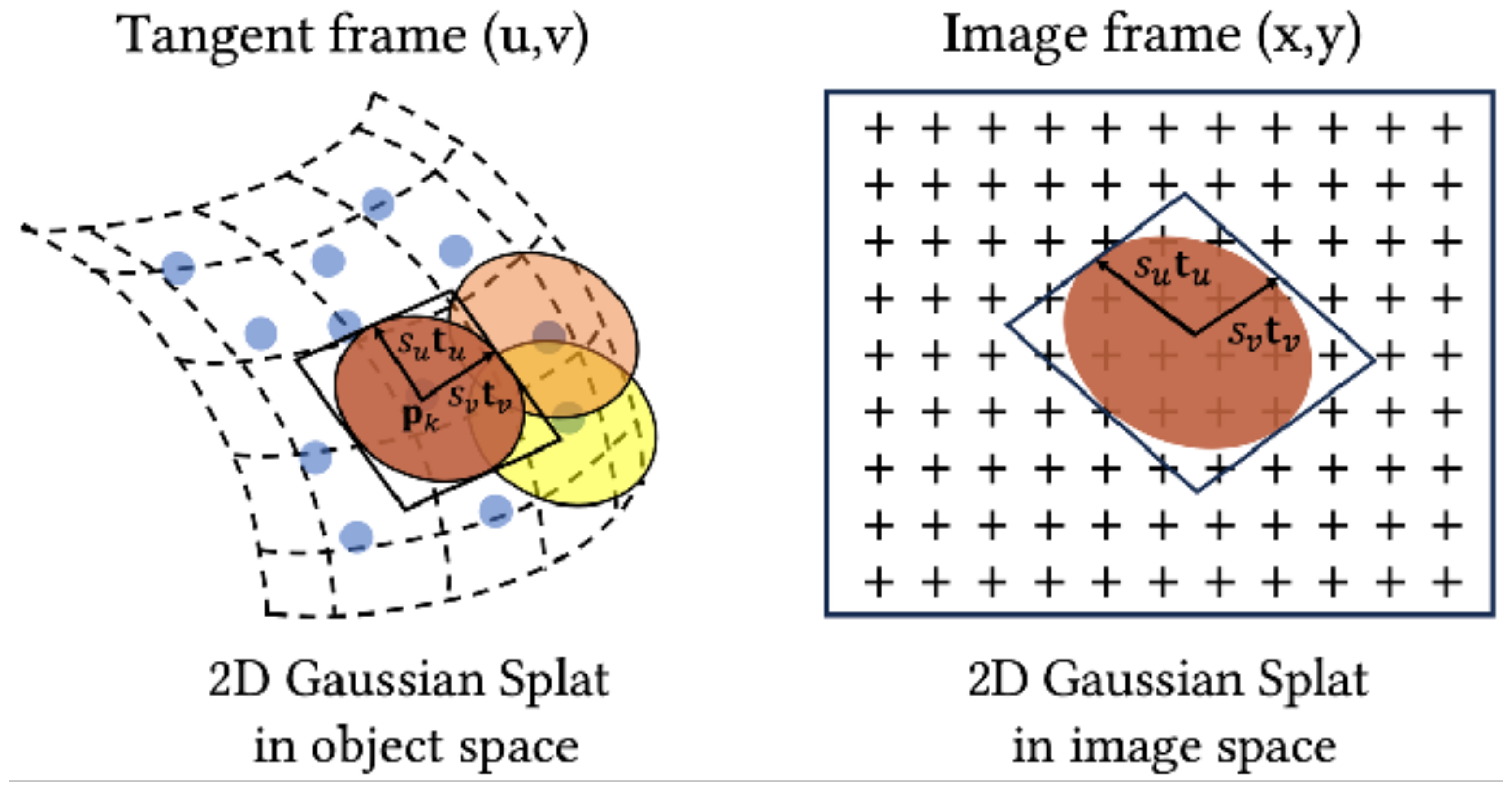


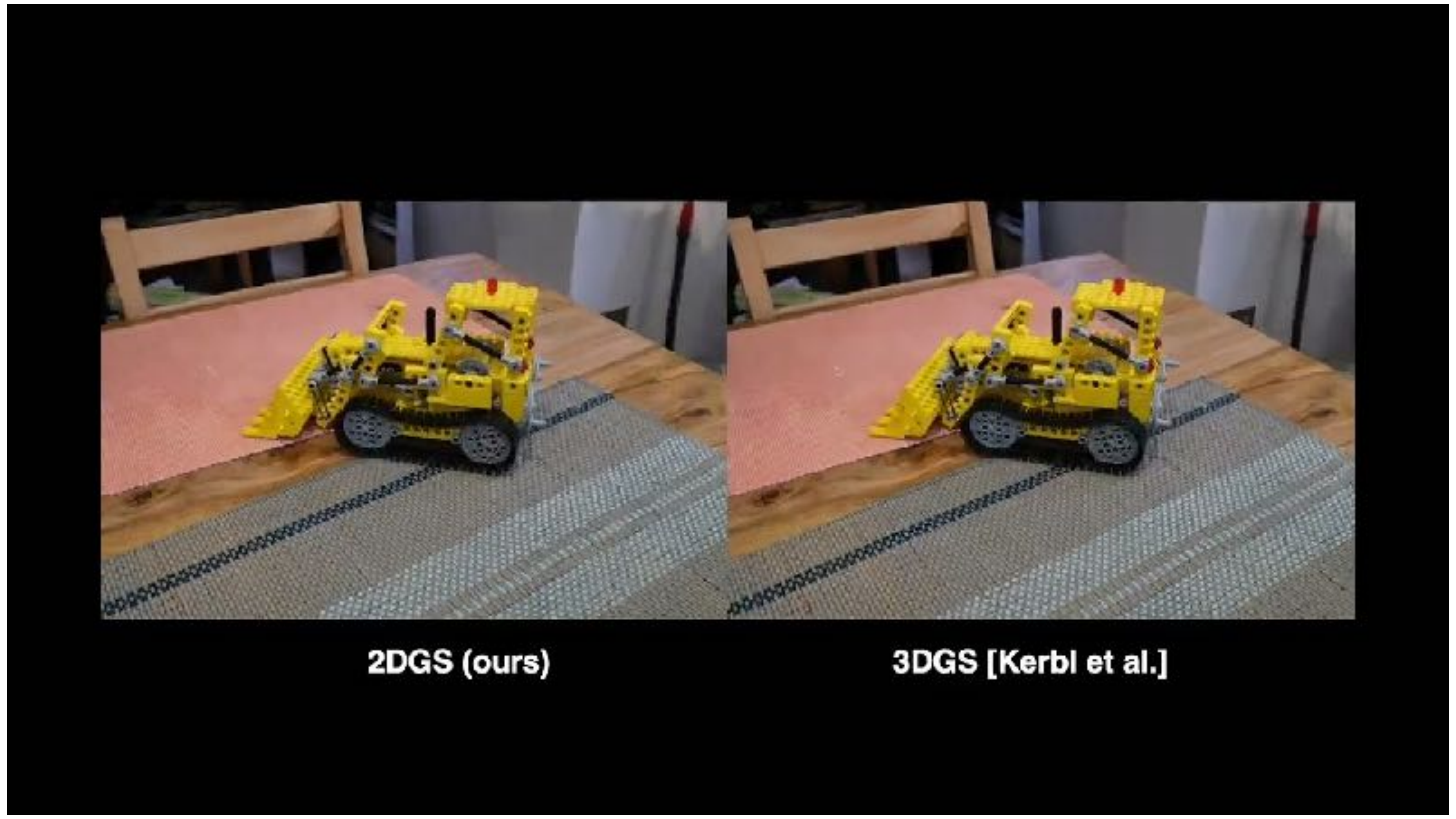


https://surfsplatting.github.io/

3D Gaussian Splatting
Variánsok — 2DGS







https://surfsplatting.github.io/

3D Gaussian Splatting

Variánsok — 2DGS — Mélység renderelés

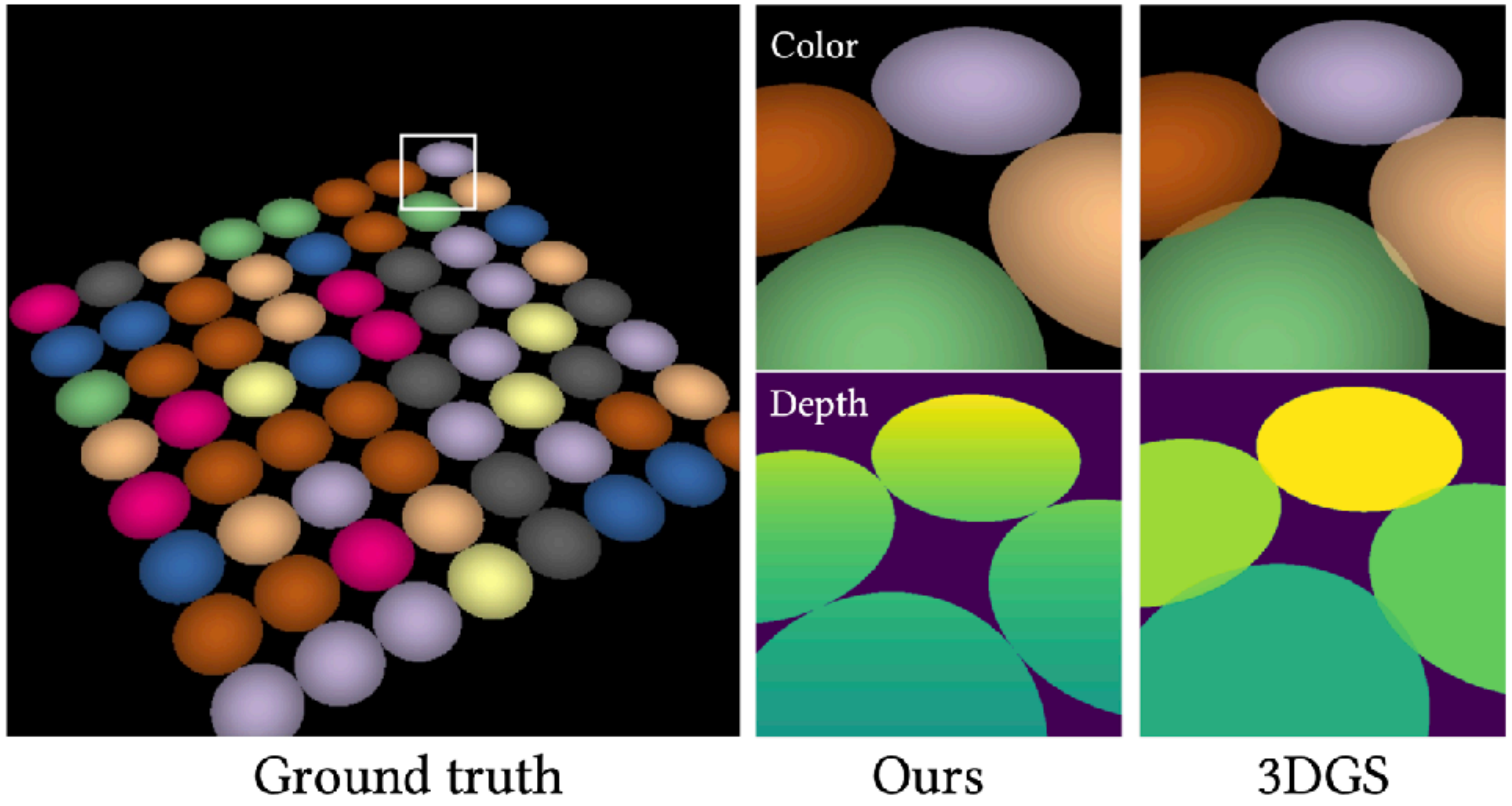
$$z_{\mathrm{mean}} = \sum_i \omega_i z_i / (\sum_i \omega_i + \epsilon) \qquad z_{\mathrm{median}} = \max\{z_i | T_i > 0.5\}$$

$$\omega_i = T_i \alpha_i \hat{\mathcal{G}}_i(\mathbf{u}(\mathbf{x}))$$

$$T_i = \prod_{j=1}^{i-1}(1 - \alpha_j \hat{\mathcal{G}}_j(\mathbf{u}(\mathbf{x})))$$

<u>Mélységkép renderelése</u>

3DGS regularizációjára is használatos!



3D Gaussian Splatting

Variánsok — 2DGS — Mesh extrakció

3DGS

2DGS

Mesh extrakció — volumetrikus fúzió + marching cubes:

3DGS → Mélységkép → SDF → Mesh

3D Gaussian Splatting

Variánsok — Markov Chain Monte Carlo (MCMC)

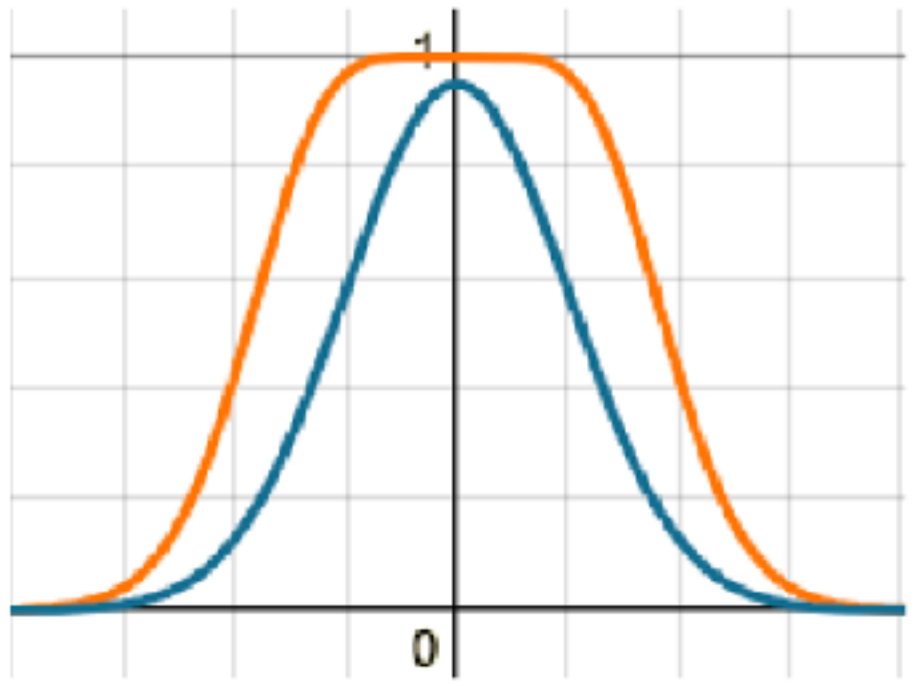
3D Gaussian Splatting as Markov Chain Monte Carlo

$$\mathbf{g} \leftarrow \mathbf{g} - \lambda_{\mathrm{lr}} \cdot \nabla_{\mathbf{g}} \mathbb{E}_{\mathbf{I} \sim \mathcal{I}} \left[\mathcal{L}_{\mathrm{total}} \left(\mathbf{g}; \mathbf{I} \right) \right] + \boxed{\lambda_{\mathrm{noise}} \cdot \boldsymbol{\epsilon}}$$

$$\boldsymbol{\epsilon} = \lambda_{\mathrm{lr}} \cdot \sigma\big(-k(o - t)\big) \cdot \mathbf{\Sigma} \eta$$

$$\eta \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

- Gradiens lépéshez random zaj hozzáadása (a la Langevin dinamika)
  - Zaj skálázása opacitás, kovariancia és learning rate alapján
- 3DGS optimalizáció heurisztikáinak újraértelmezése a Markov láncok nyelvén
- Sokkal jobb eredmények random inicializációval!
- A legtöbb modern 3DGS könyvtár implementálja



3DGS

MCMC

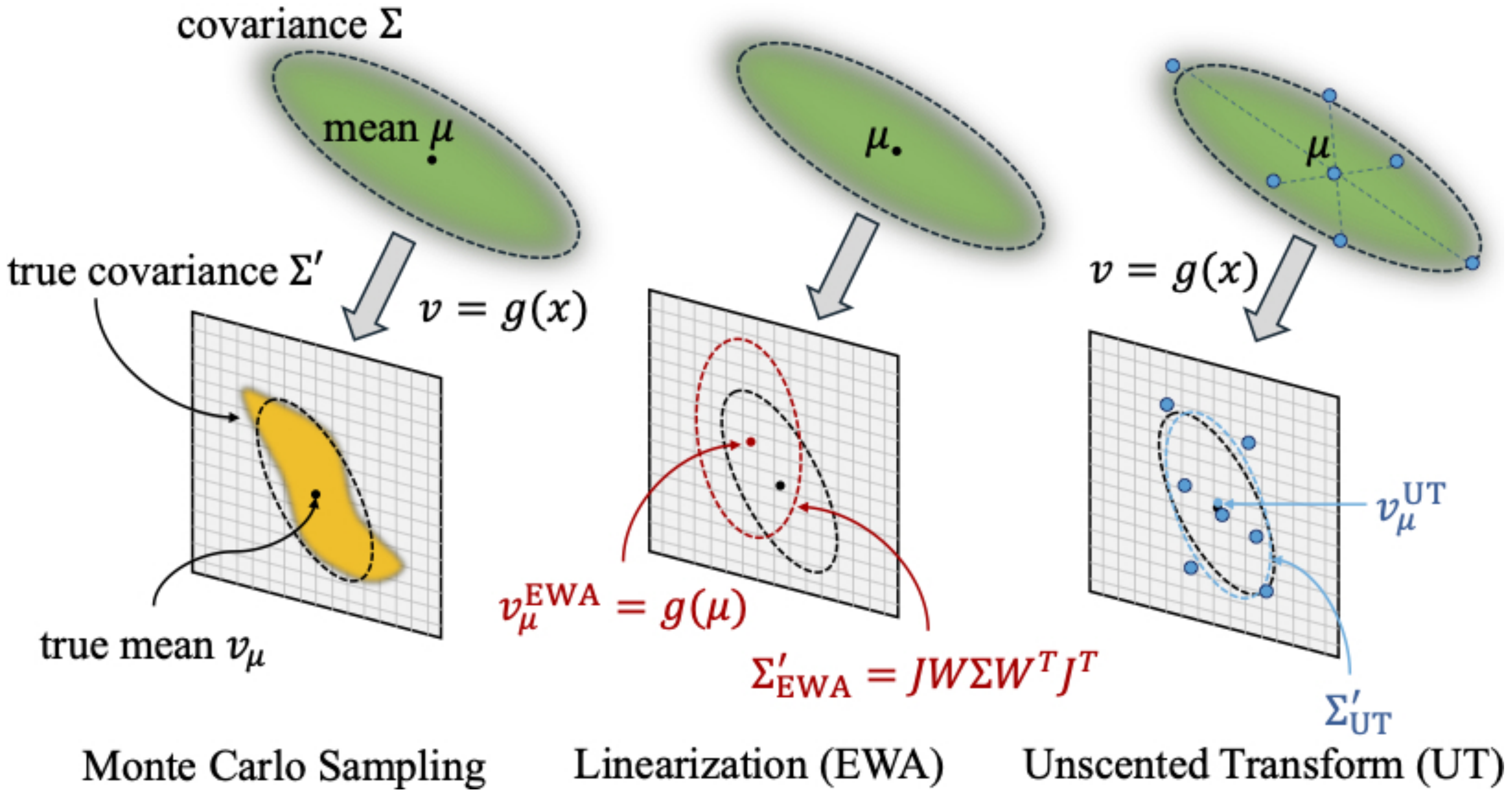
Gauss “klónozás” előtt vs. utána

https://ubc-vision.github.io/3dgs-mcmc/

3D Gaussian Splatting

Variánsok — 3DGUT

<u>Általános (valóságos) kameramodellek támogatása!</u>

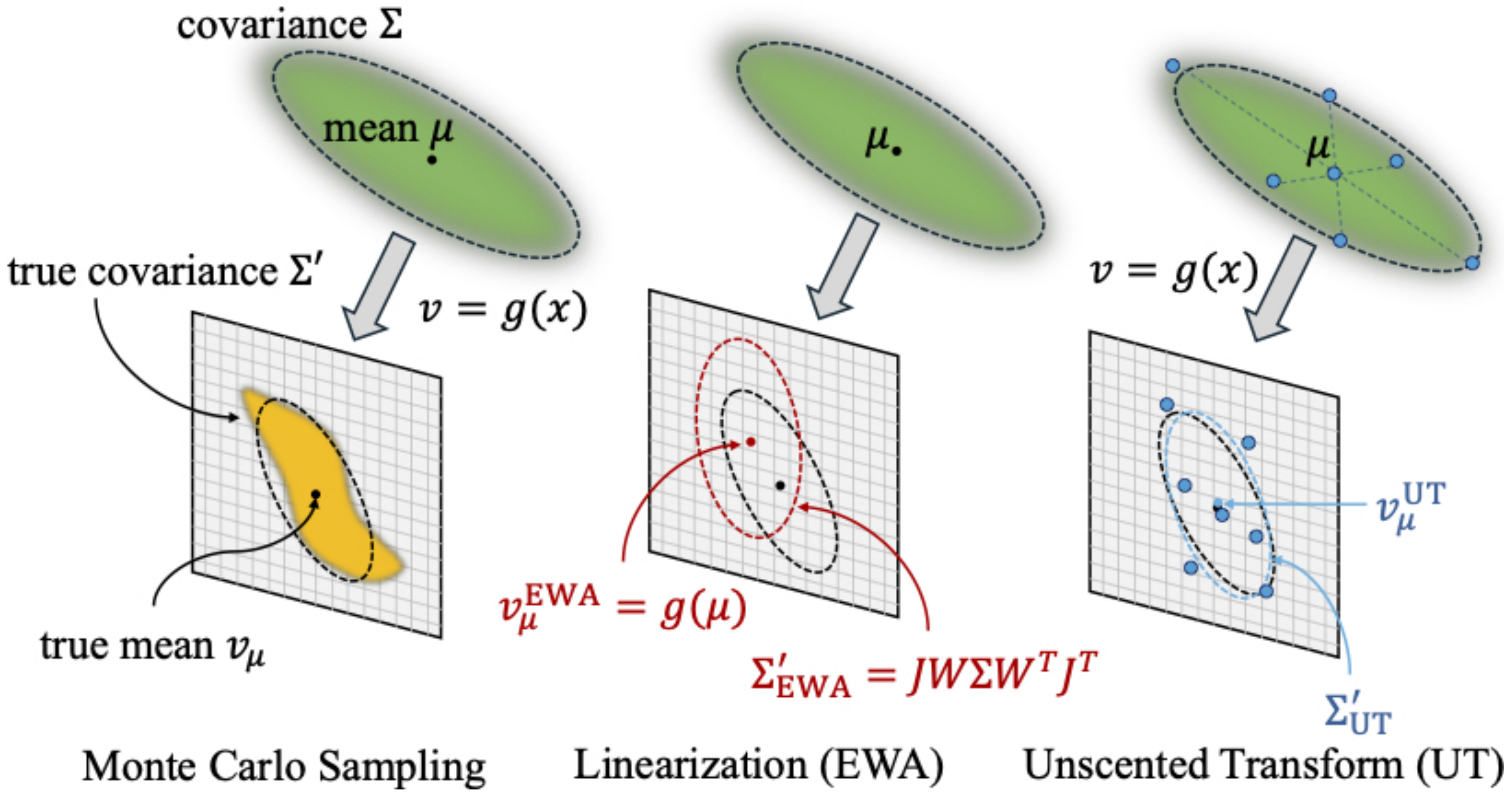


https://research.nvidia.com/labs/toronto-ai/3DGUT/

3D Gaussian Splatting

Variánsok — 3DGUT

Általános (valóságos) kameramodellek támogatása!



https://research.nvidia.com/labs/toronto-ai/3DGUT/

3D Gaussian Splatting

Variánsok — Korrekt renderelés

Does 3D Gaussian Splatting Need Accurate Volumetric Rendering?

A. Celarek[1], G. Kopanas[2,3,4], G. Drettakis[3,4], M. Wimmer[1] and B. Kerbl[1]

(a) *Viewing Scenario* **(b)** *Per-Gaussian sorting (3DGS)* **(c)** *Per-pixel sorting (STP)* **(d)** *Exact integration (ray marcher)*

A globális Gauss sortolás "csalás"…
alternatívák:
per-pixel sort (StopThePop)
vagy
rendes volume rendering (ray marching)

3D Gaussian Splatting
Variánsok — Korrekt renderelés

Does 3D Gaussian Splatting Need Accurate Volumetric Rendering?

A. Celarek[1], G. Kopanas[2,3,4], G. Drettakis[3,4], M. Wimmer[1] and B. Kerbl[1]

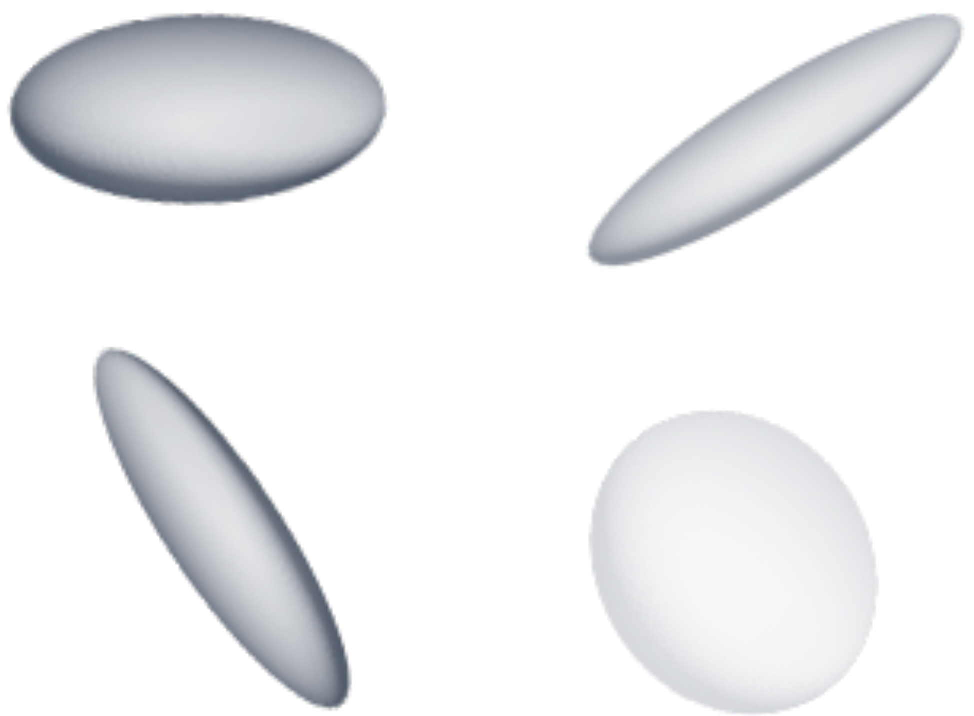
Kevés Gauss-szal még látszik valami különbség… több Gauss-szal már nem!

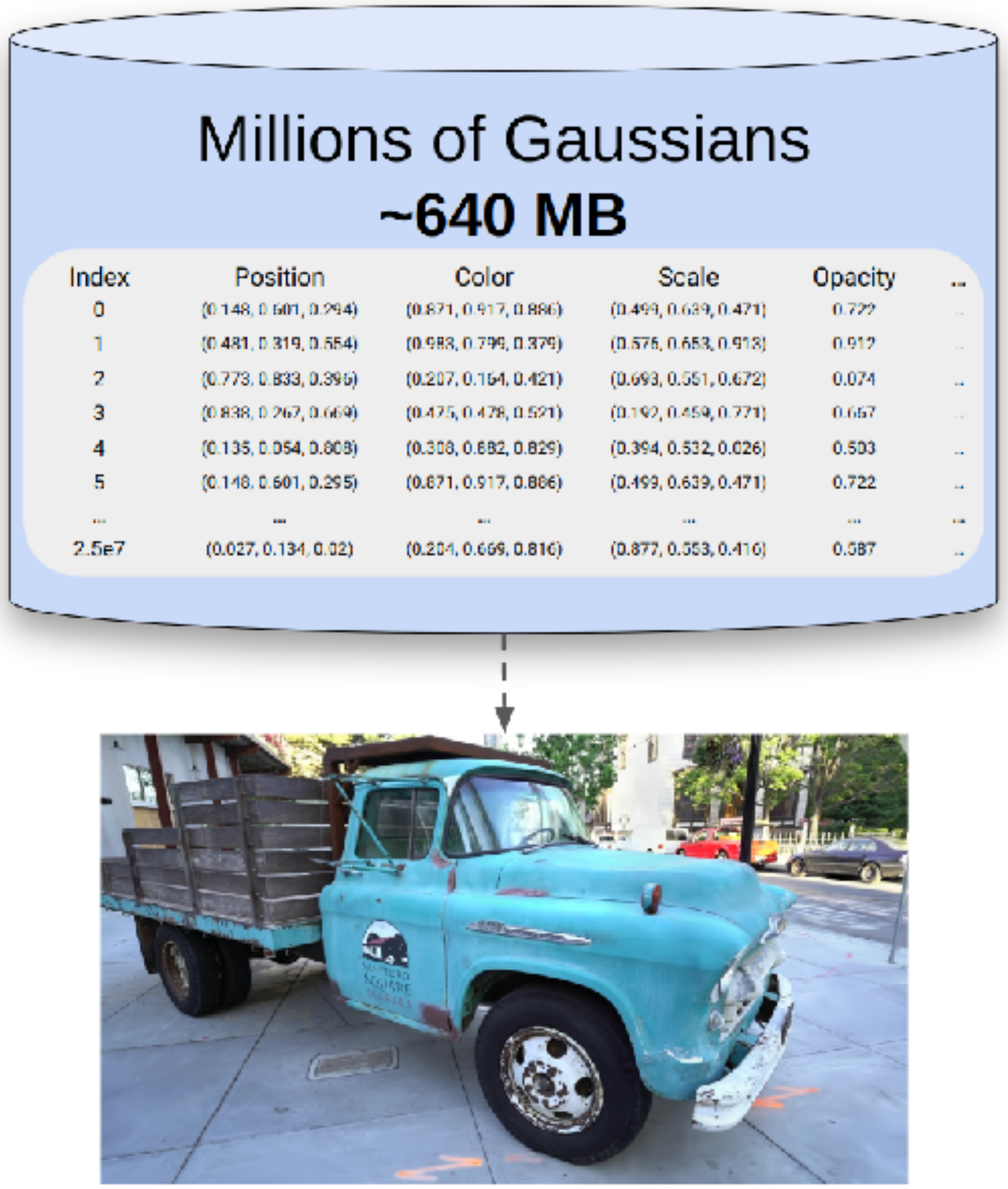
A “bitter lesson” újabb példája:
kellően sok paraméterrel a fizikai korrektség kevésbé lényeges,
mint a *modell rugalmassága* és az *optimalizáció hatékonysága*!

3D Gaussian Splatting

Tömörítés

- Gauss attribútumok:
  - 3 koordináta
  - 3 kovarianca skála
  - 4 forgatás (kvaternió)
  - 1 opacities
  - 3 RGB (diffúz szín, 0. rendű gömbi harmonikus együttható)
  - 45 gömbi harmonikus együttható (1.-3. rendű gömbi harmonikus, nézetfüggő színezés)
- 59 float = 236 byte minden Gauss tárolásához!
- 1 Millió Gauss: > 200 MB memóriafoglalás — lehet tömöríteni??





Index	Position	Color	Scale	Opacity	...
0	(0.148, 0.601, 0.294)	(0.871, 0.917, 0.886)	(0.499, 0.639, 0.471)	0.722	...
1	(0.481, 0.319, 0.554)	(0.983, 0.799, 0.379)	(0.576, 0.653, 0.913)	0.912	...
2	(0.773, 0.833, 0.396)	(0.207, 0.164, 0.421)	(0.693, 0.551, 0.672)	0.074	...
3	(0.838, 0.267, 0.669)	(0.475, 0.478, 0.521)	(0.192, 0.459, 0.771)	0.667	...
4	(0.135, 0.054, 0.808)	(0.308, 0.882, 0.829)	(0.394, 0.532, 0.026)	0.503	...
5	(0.148, 0.601, 0.295)	(0.871, 0.917, 0.886)	(0.499, 0.639, 0.471)	0.722	...
...	...	...	...	...	...
2.5e7	(0.027, 0.134, 0.02)	(0.204, 0.669, 0.816)	(0.877, 0.553, 0.416)	0.587	...

PSNR 25.44dB

3D Gaussian Splatting

Tömörítés — SPZ formátum

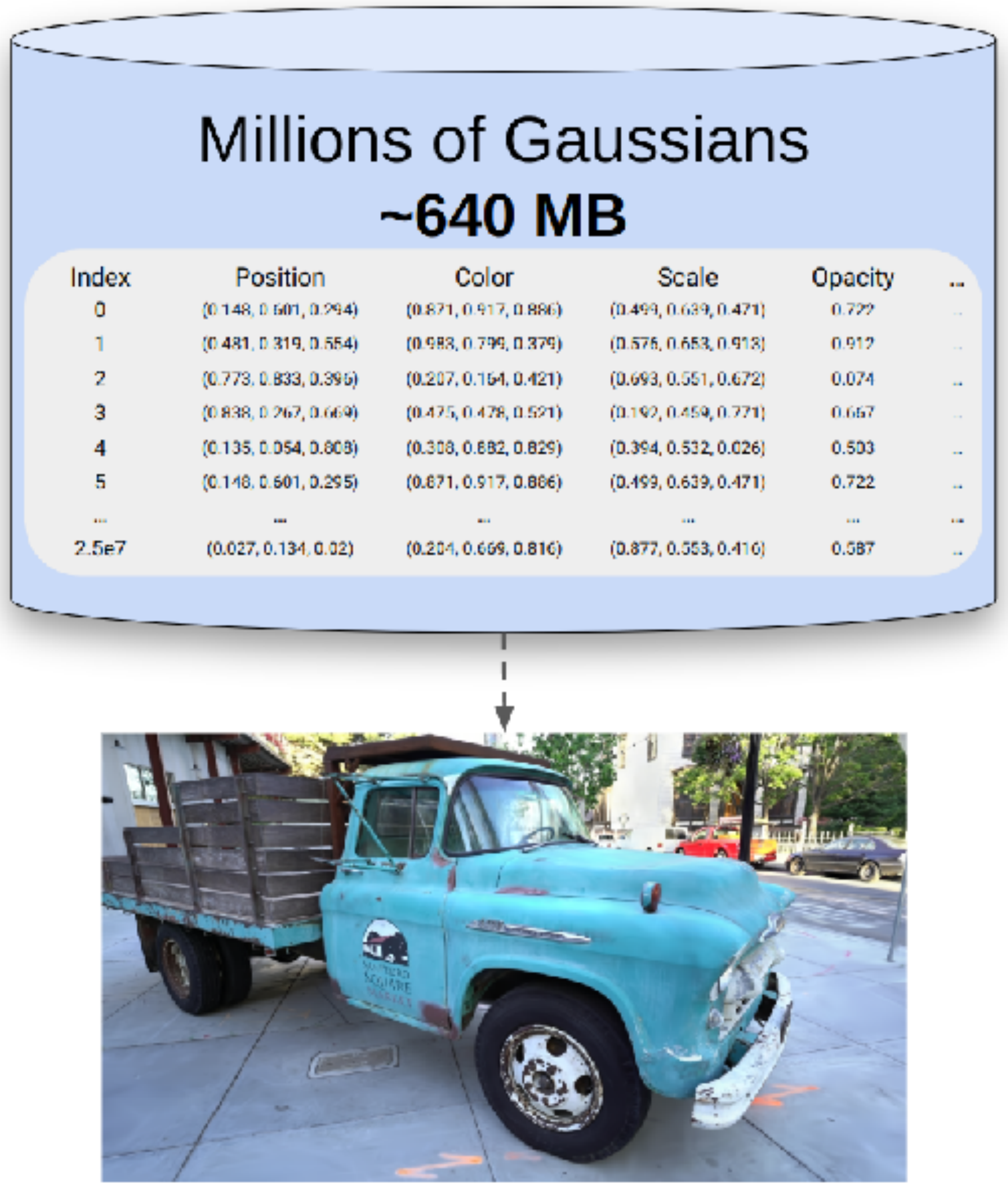
- Niantic SPZ — “JPG for 3DGS”
- Veszteséges tömörítés
- Attribútumok kvantálása a minőség megőrzése mellett
- Hatékony párhuzamosított ZIP tömörítés (ZSTD)
- Nyílt forráskódú implementáció
- Széles körben adoptált átviteli formátum
  - Pl. a GLTF extension is használja

Element	SPZ Format	PLY Format	Size Reduction (%)
Positions	24-bit fixed point integer with adjustable fractional bits	32-bit or 64-bit floating-point	25%–62.5%
Rotation	3 components of a quaternion stored as 8-bit signed integers	4 components of quaternion as 32-bit floats	81.25%
Colors (RGB)	8-bit unsigned integers per channel	Typically 8-bit or 32-bit floats per channel	0%–75%
Scales	8-bit log-encoded integer	Typically 32-bit or 64-bit floating-point	75%–87.5%
Alphas (Transparency)	8-bit unsigned integer	Typically 32-bit float	75%
Spherical Harmonics	8-bit signed integers for coefficients, with 4-5 bits of precision	Varies, but usually stored with higher precision (e.g., 32-bit floats)	75%–87.5%

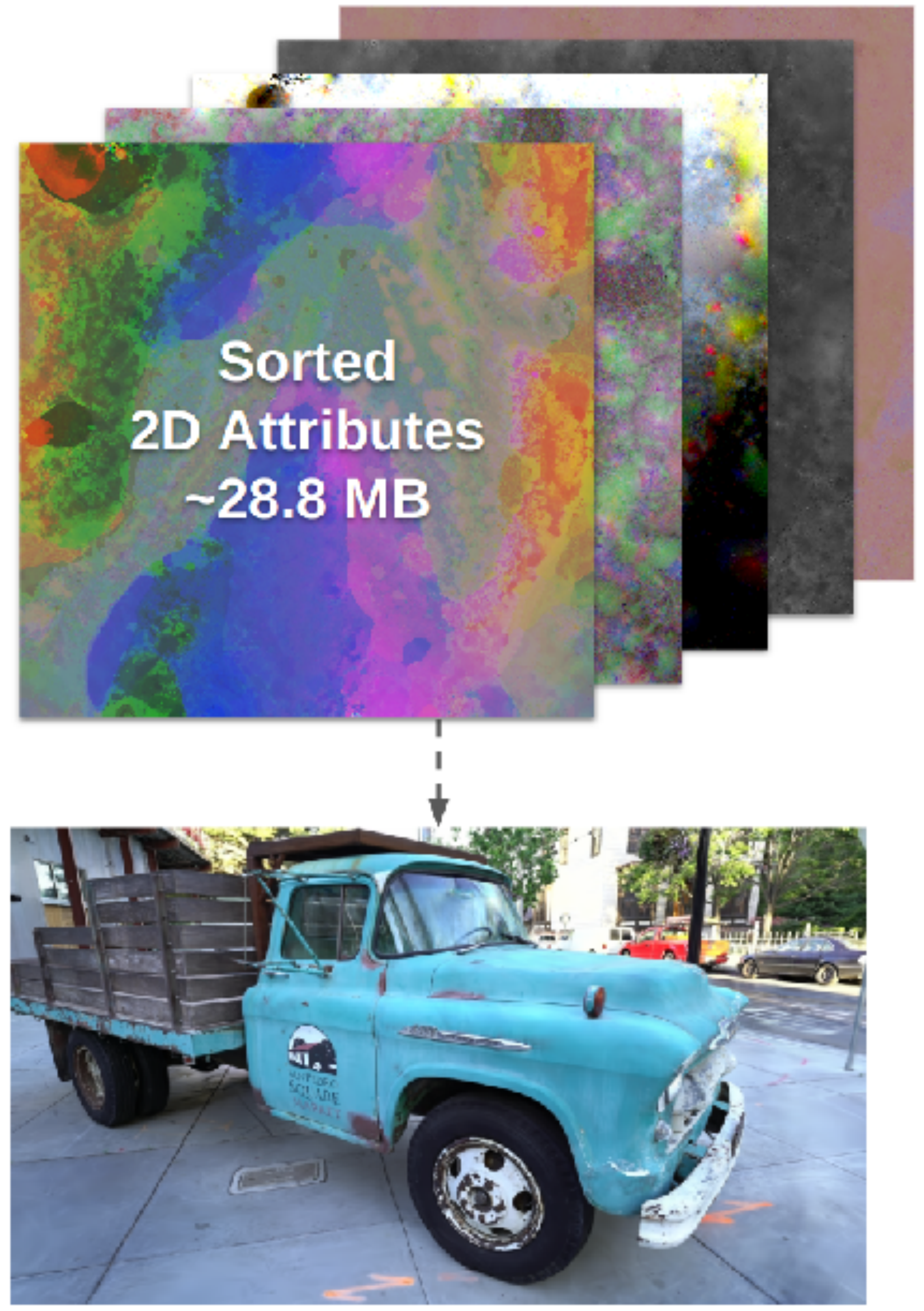
https://github.com/nianticlabs/spz

3D Gaussian Splatting

Tömörítés — Self-Organizing Gaussian (SOGS)



PSNR 25.44dB

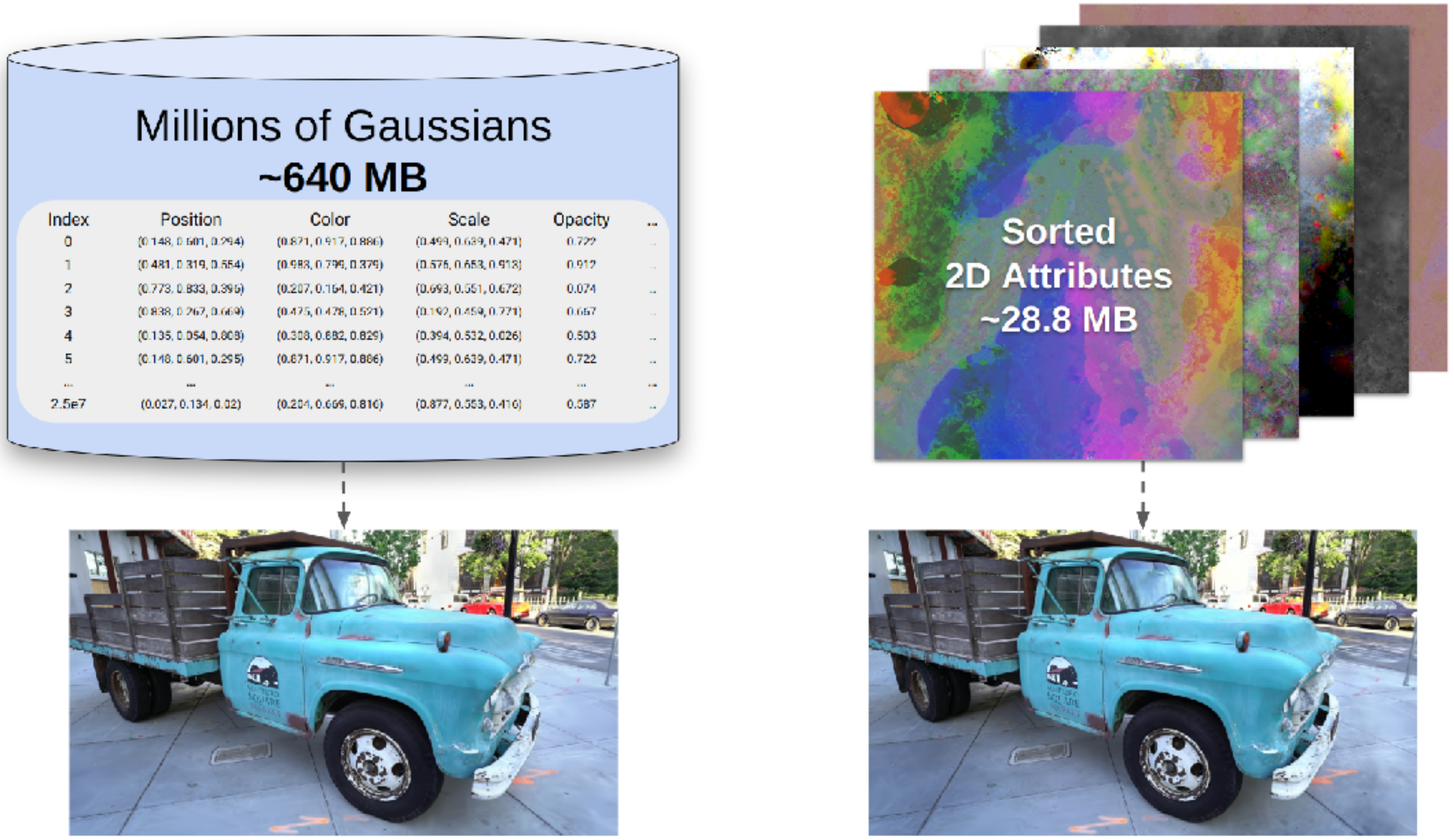


PSNR 25.57dB

https://fraunhoferhhi.github.io/Self-Organizing-Gaussians/

3D Gaussian Splatting

Tömörítés — Self-Organizing Gaussian (SOGS)



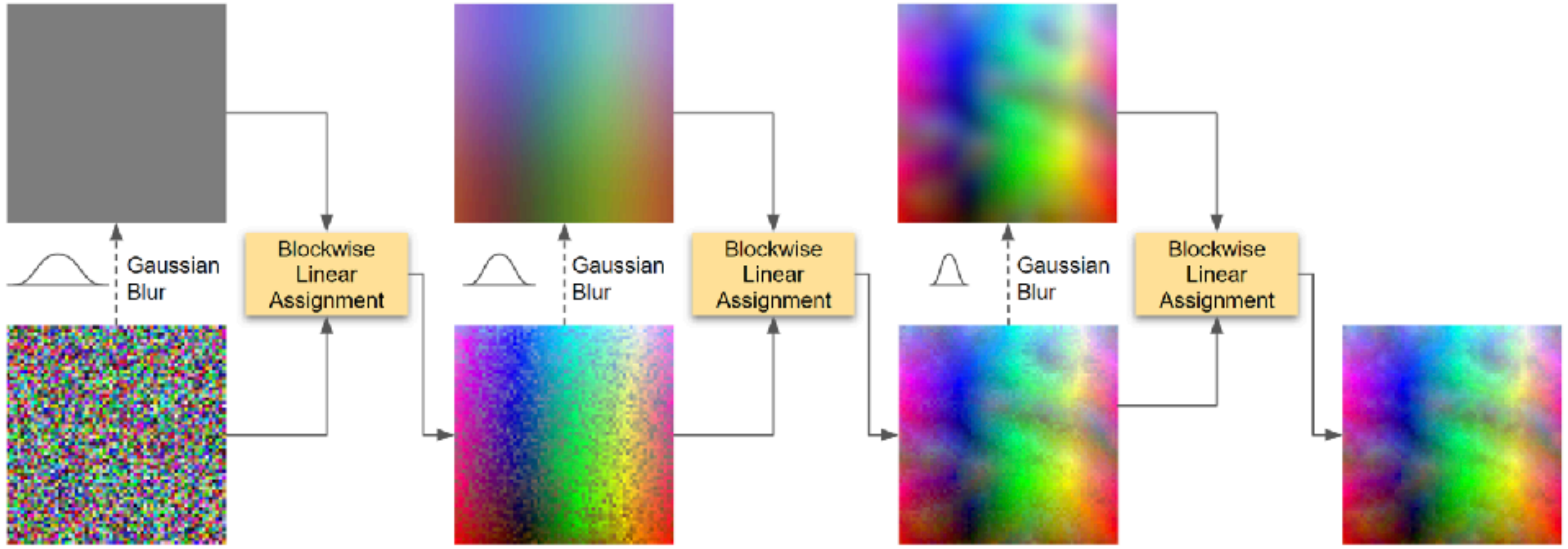
PSNR 25.44dB

PSNR 25.57dB

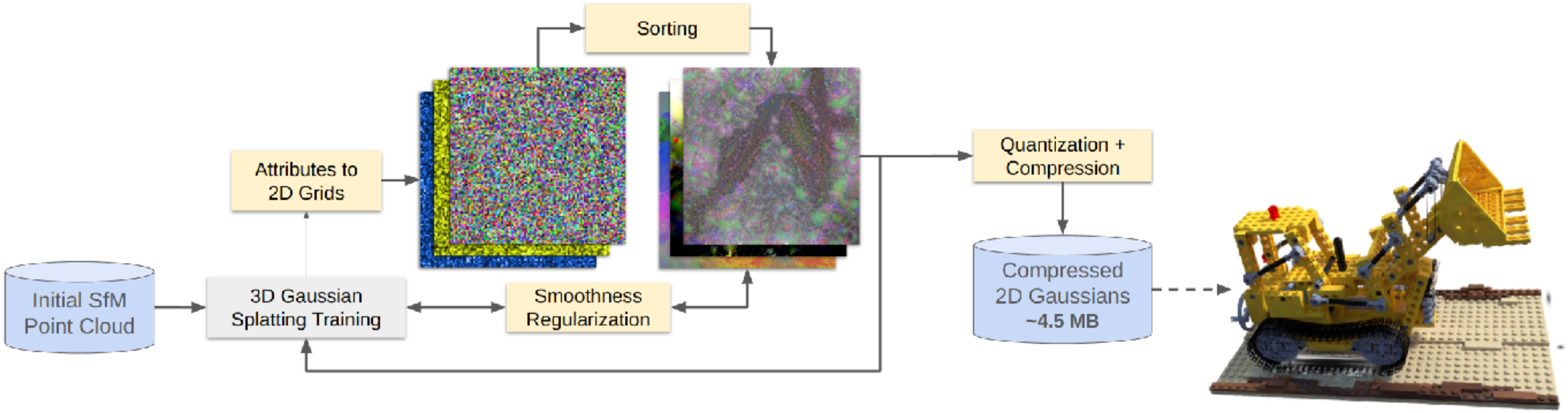
https://fraunhoferhhi.github.io/Self-Organizing-Gaussians/

3D Gaussian Splatting

Tömörítés — Self-Organizing Gaussian (SOGS)



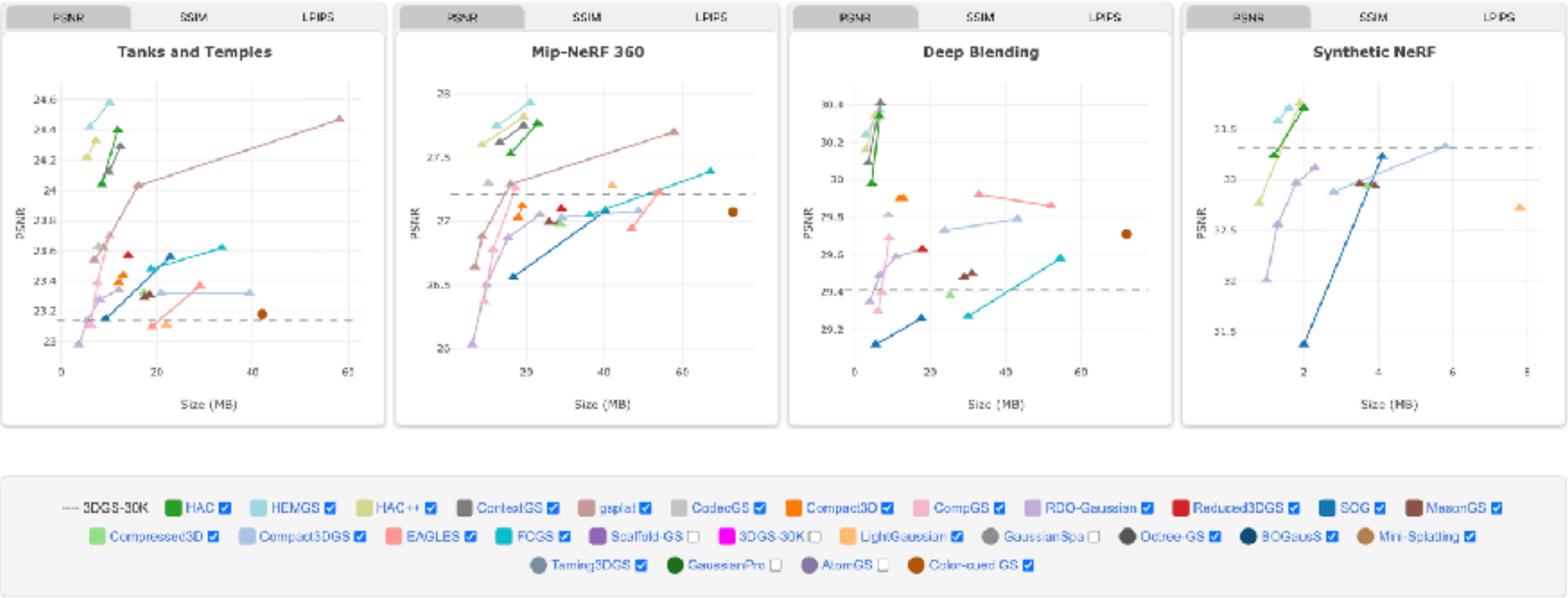
Kép pixelek speciális, hatékony sortolása



3D Gaussian Splatting

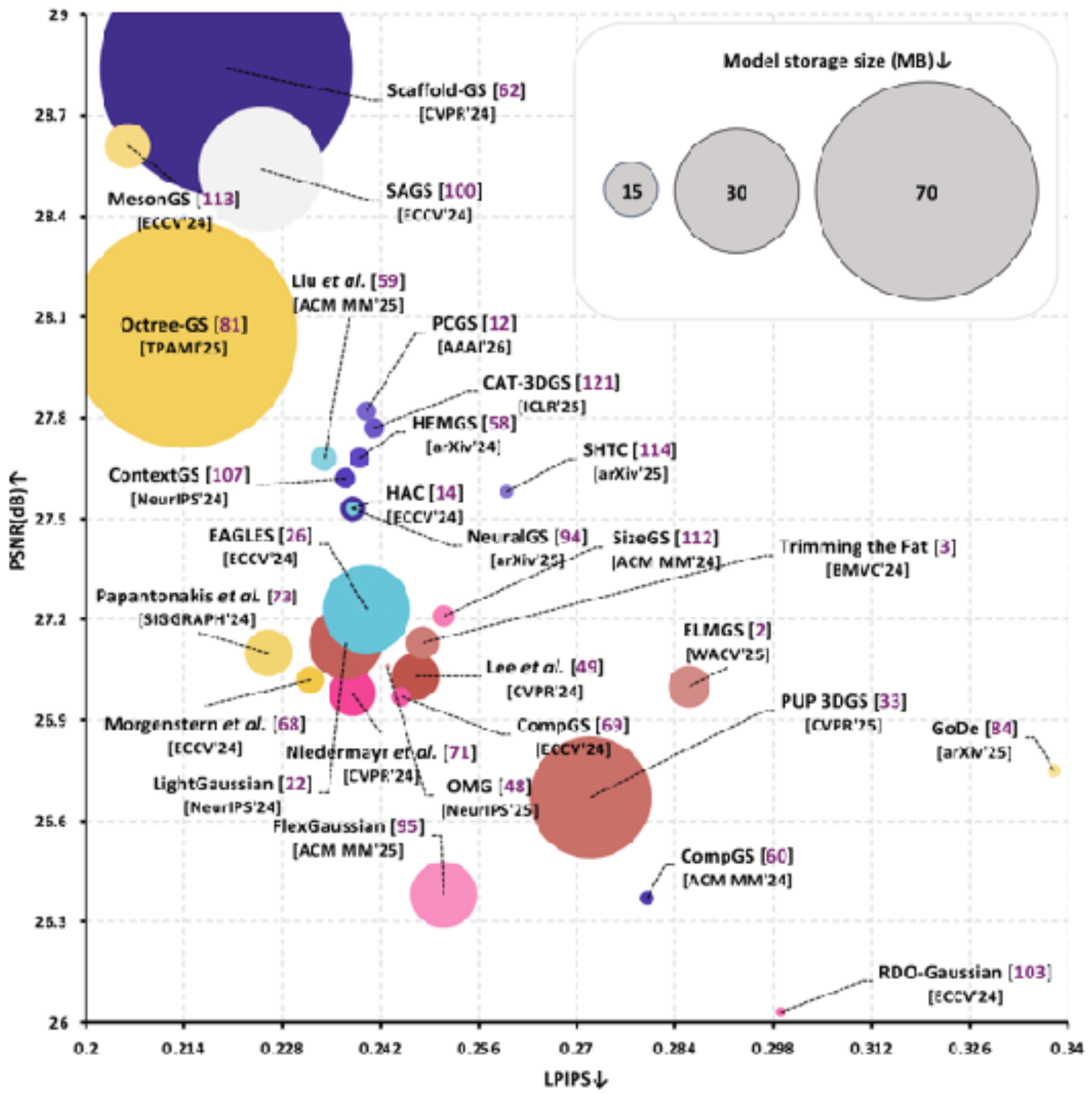
Tömörítés — Áttekintések

3DGS.zip: A survey on 3D Gaussian Splatting Compression Methods



https://w-m.github.io/3dgs-compression-survey/

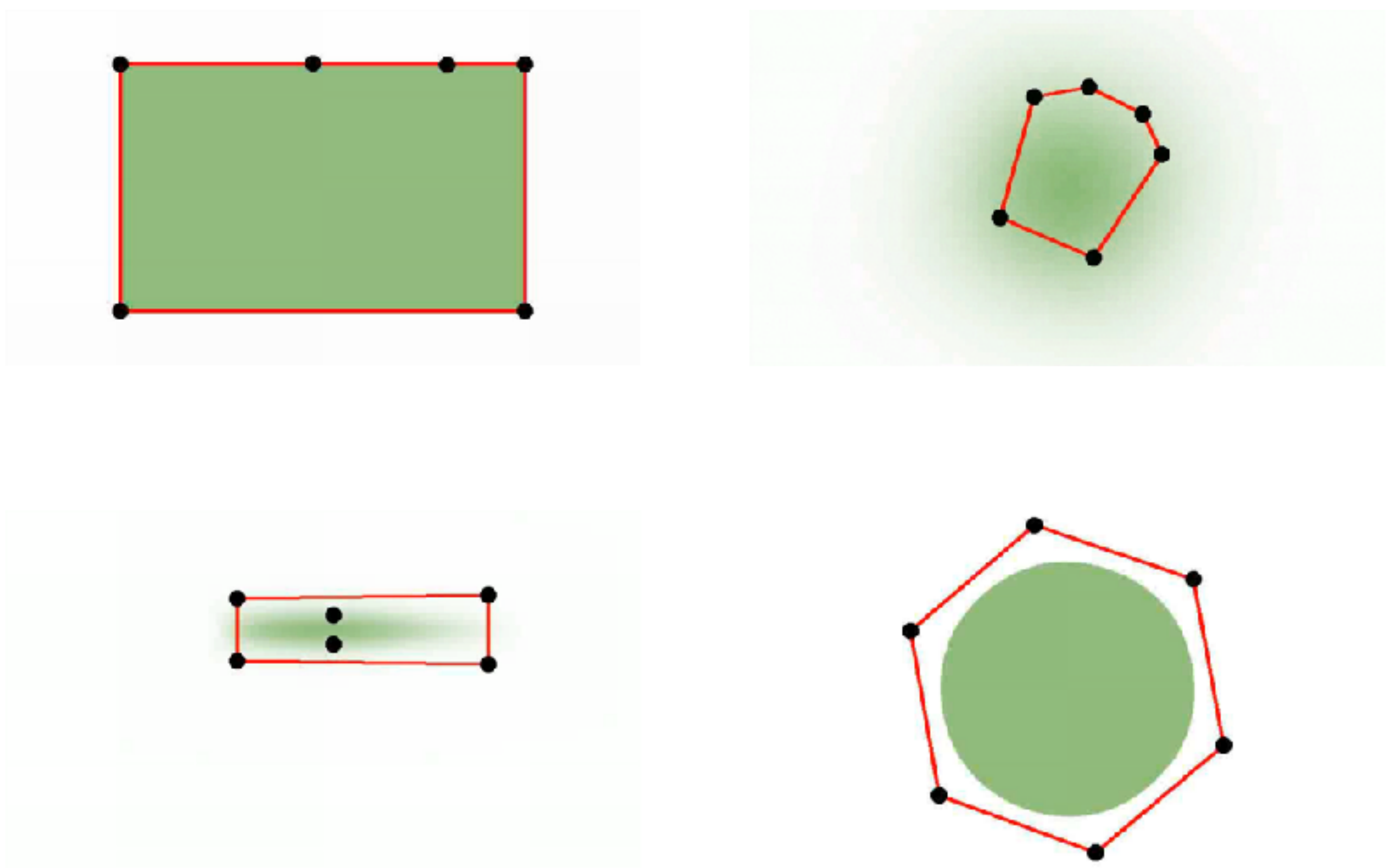
🏆 SUCCESS-GS: Survey of Compactness and Compression for Efficient Static and Dynamic Gaussian Splatting



https://cmlab-korea.github.io/Awesome-Efficient-GS/

További 3DGS Variánsok

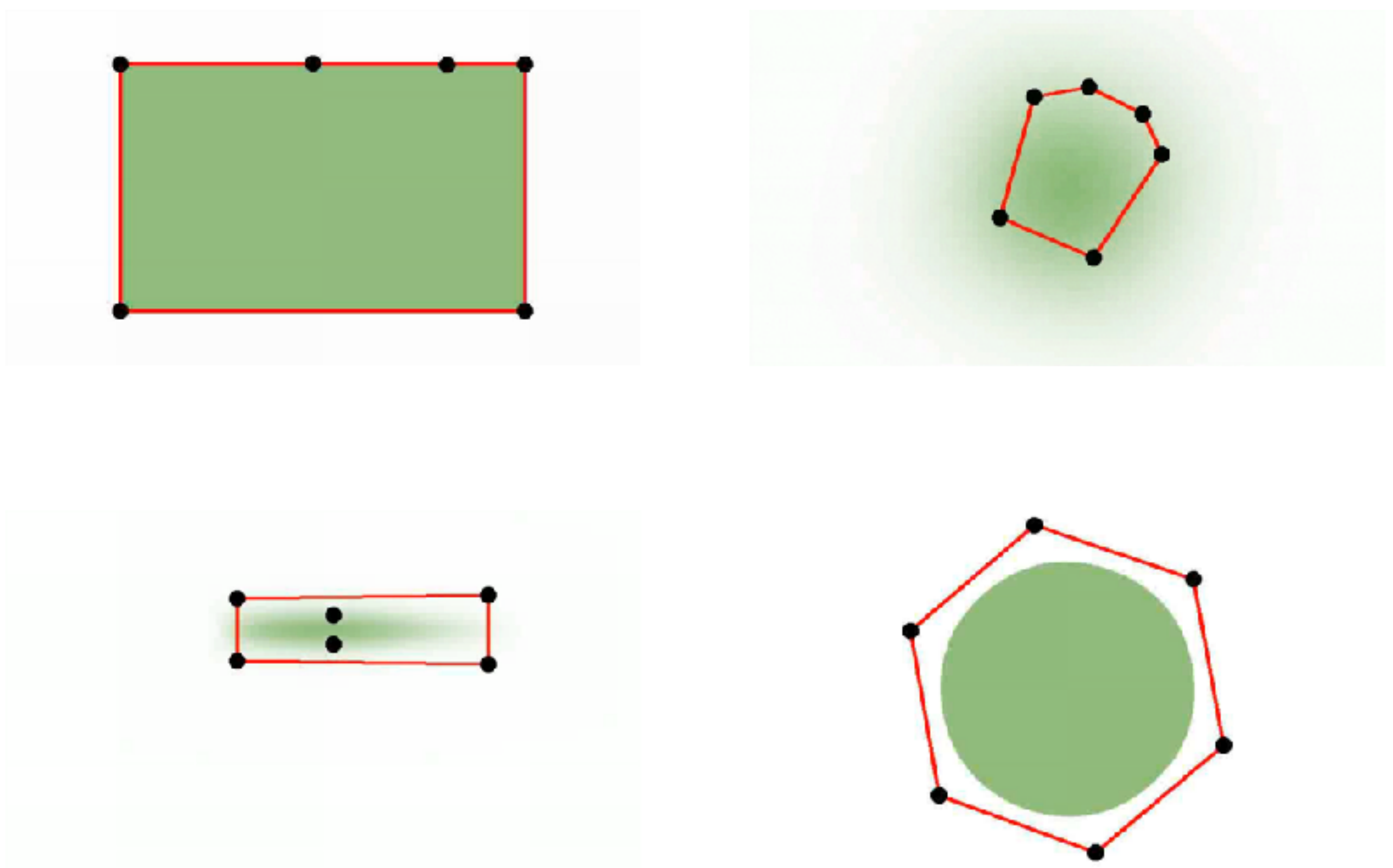
Convex Splatting



https://convexsplatting.github.io/

További 3DGS Variánsok

Convex Splatting



https://convexsplatting.github.io/

További 3DGS Variánsok

Triangle Splatting

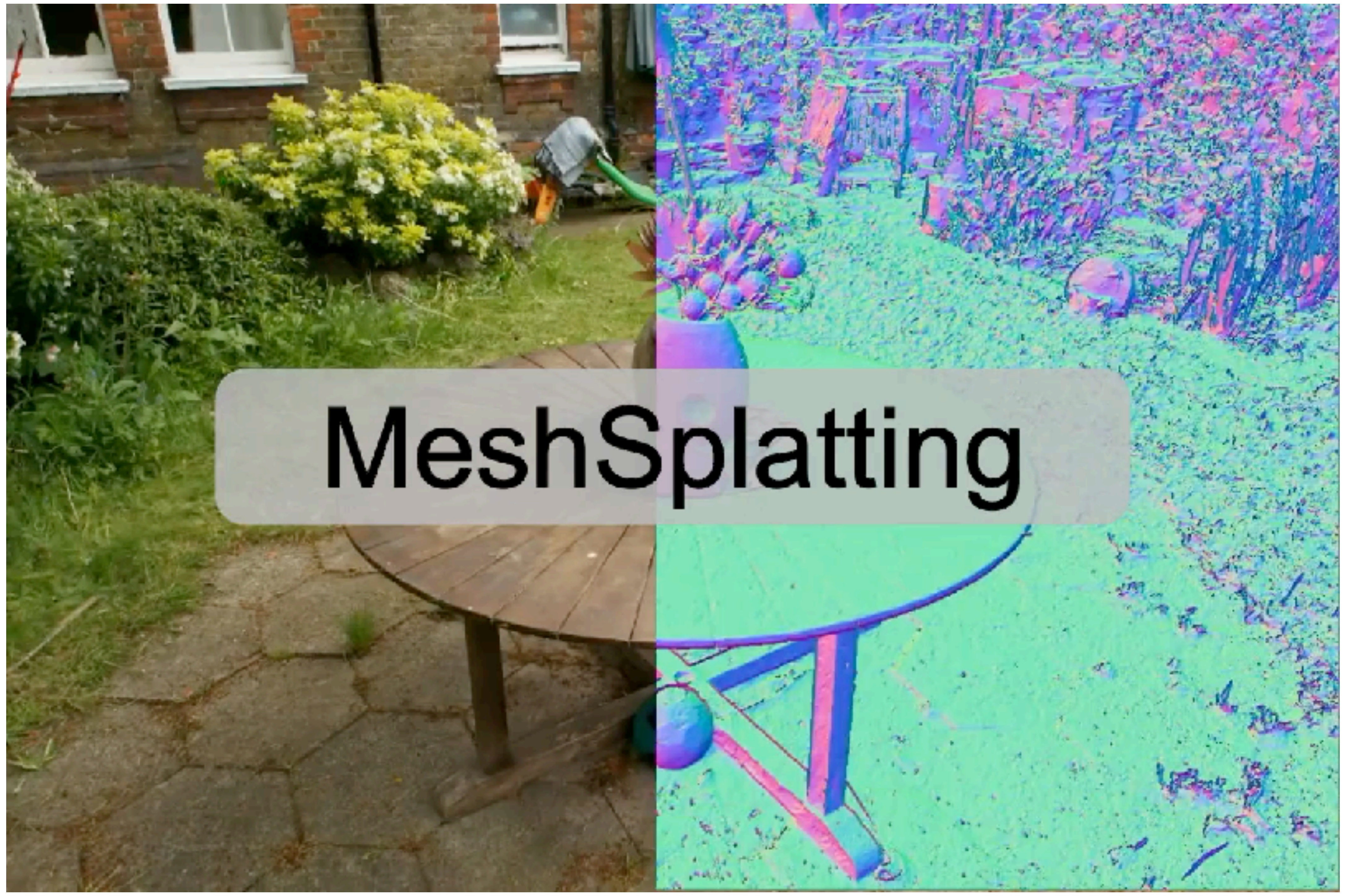
https://trianglesplatting.github.io/

További 3DGS Variánsok

Triangle Splatting

https://trianglesplatting.github.io/

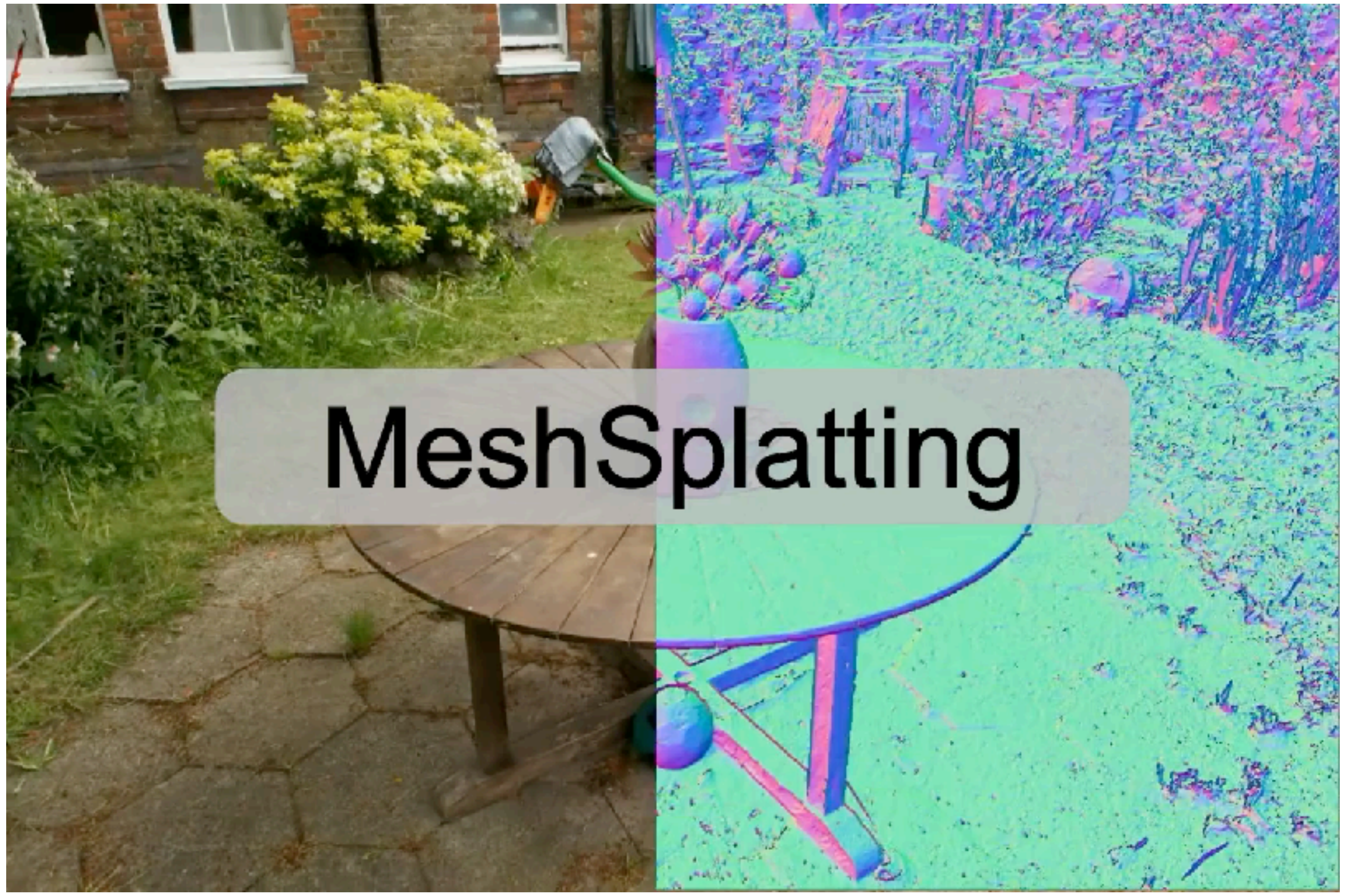
További 3DGS Variánsok
Mesh Splatting



https://meshsplatting.github.io/

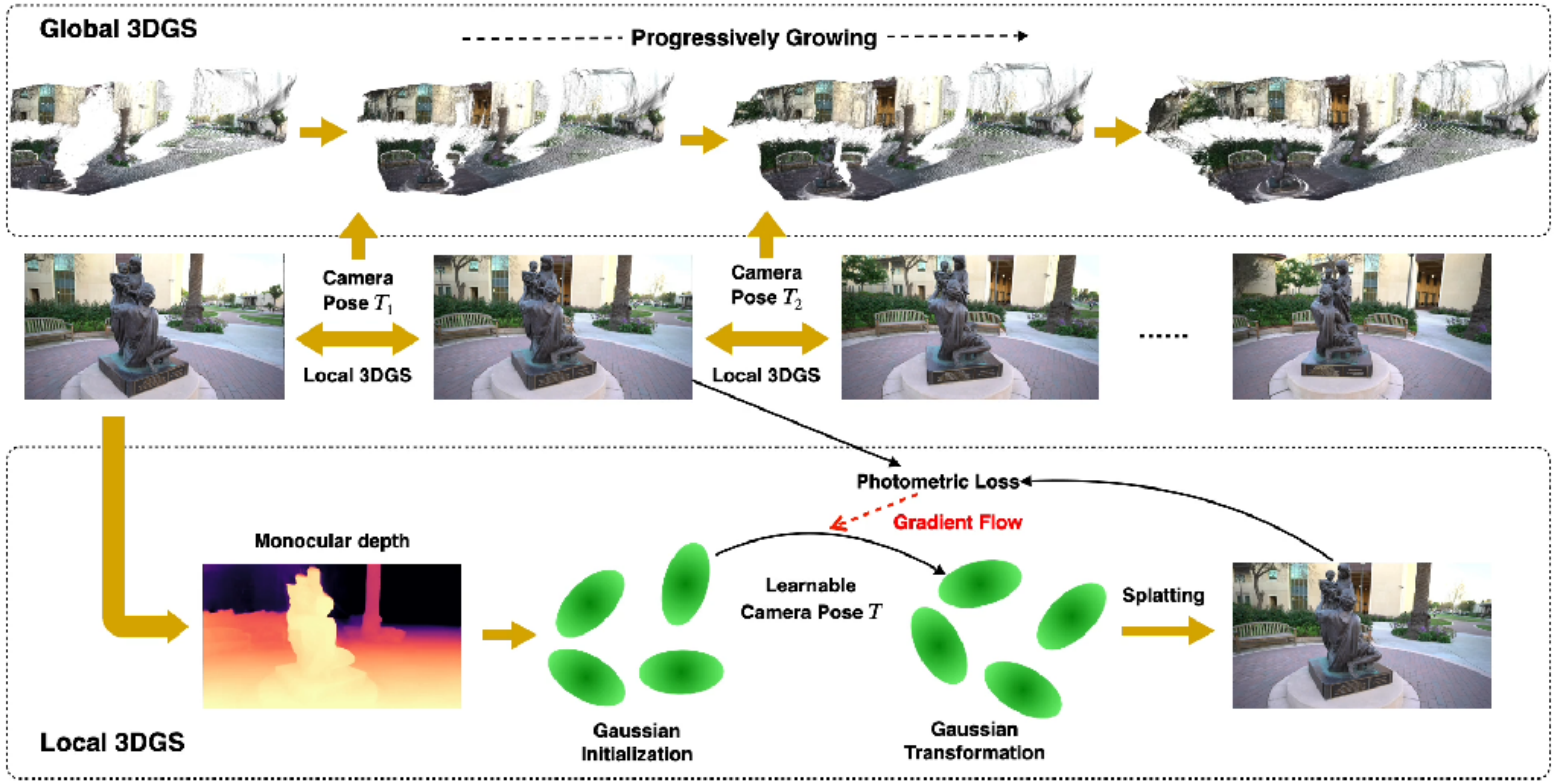
További 3DGS Variánsok

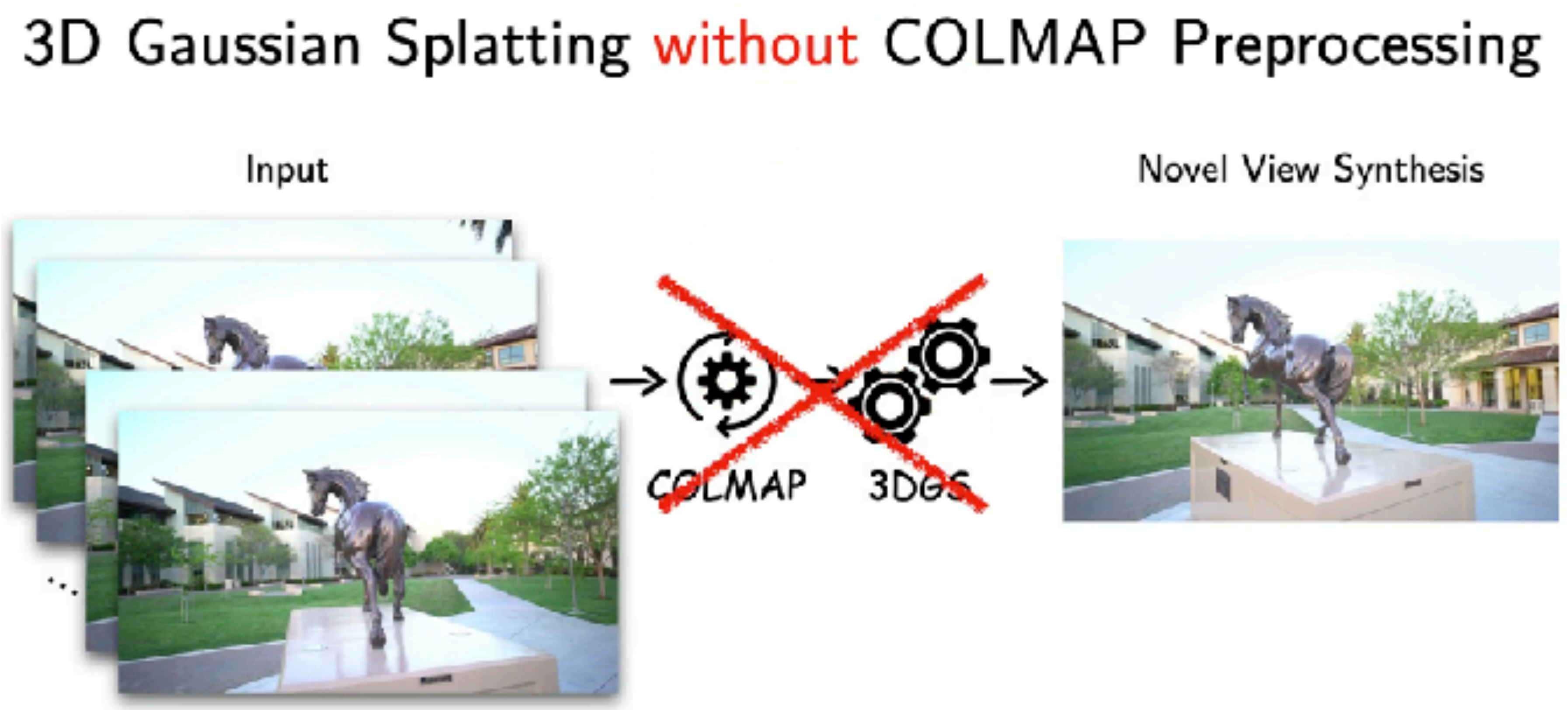
Mesh Splatting



https://meshsplatting.github.io/

További 3DGS Variánsok
COLMAP-Free 3DGS

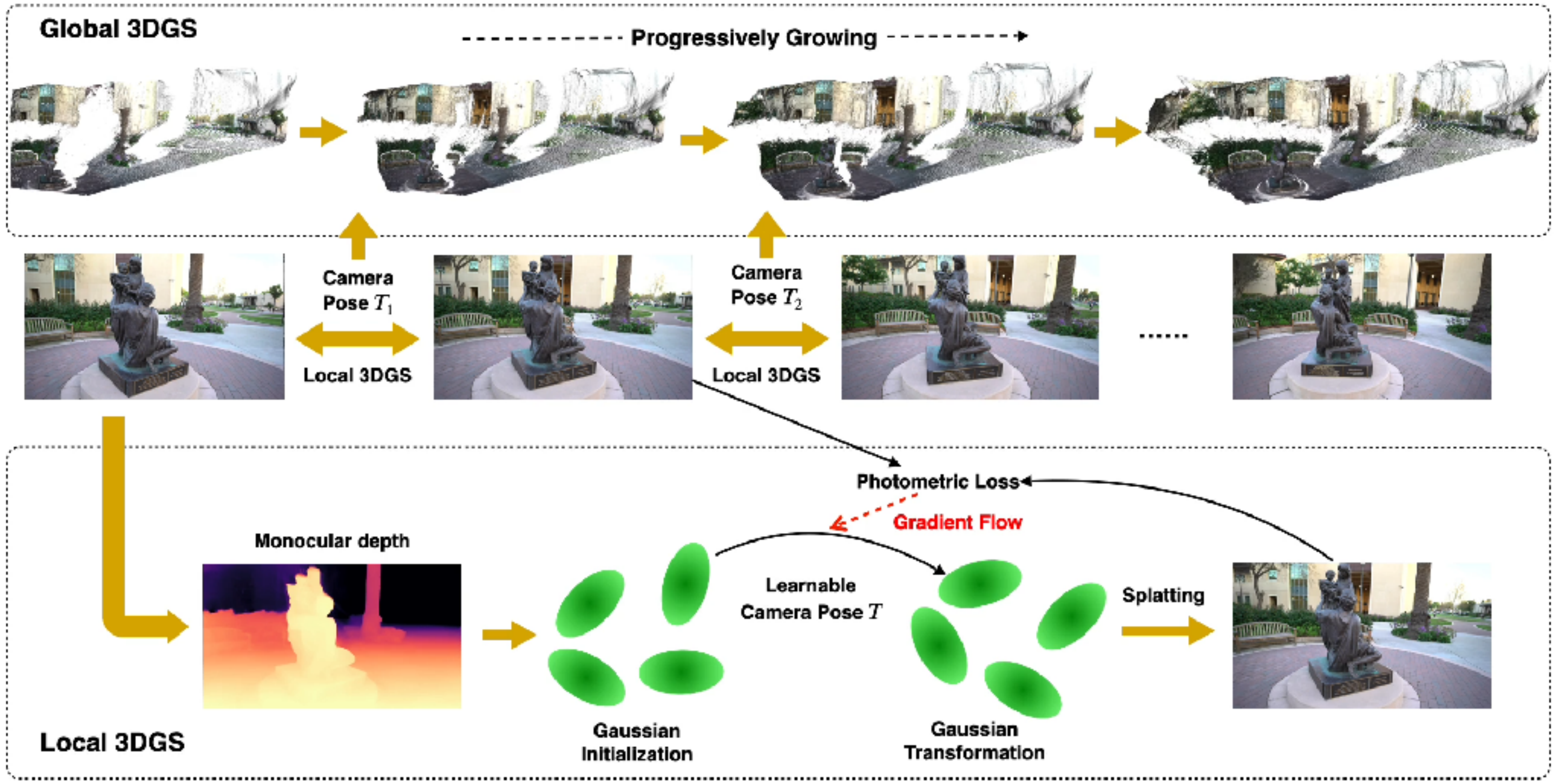


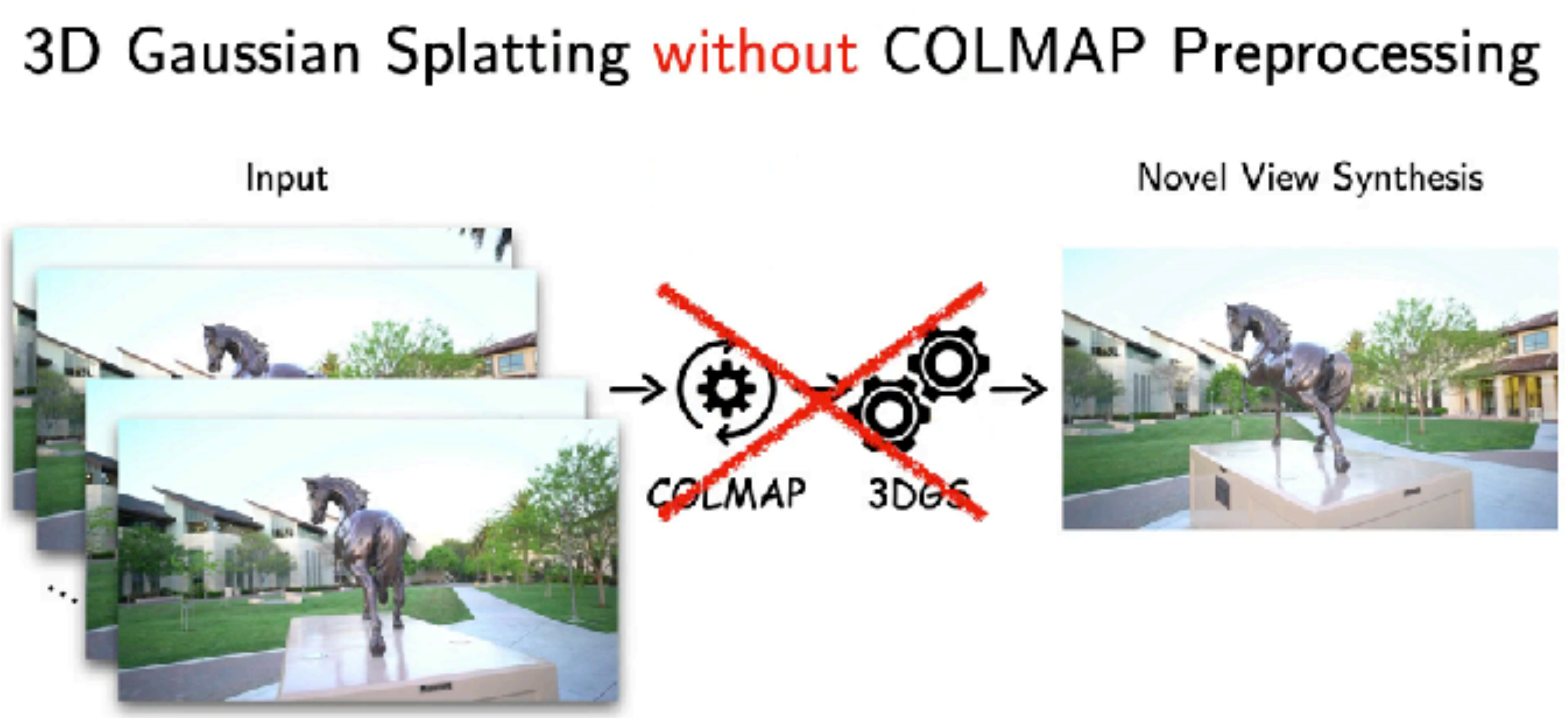


https://oasisyang.github.io/colmap-free-3dgs/

További 3DGS Variánsok

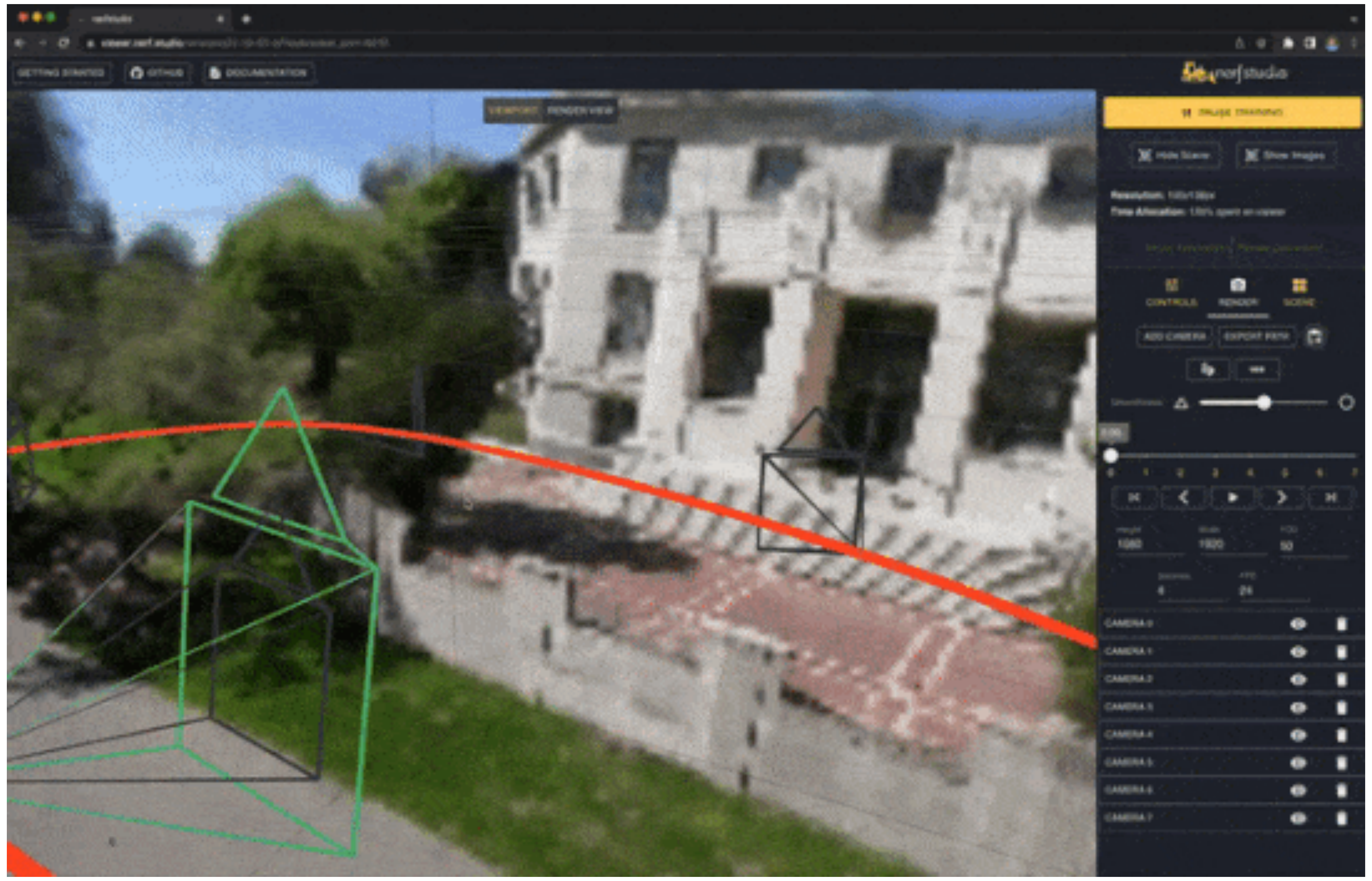
COLMAP-Free 3DGS





https://oasisyang.github.io/colmap-free-3dgs/

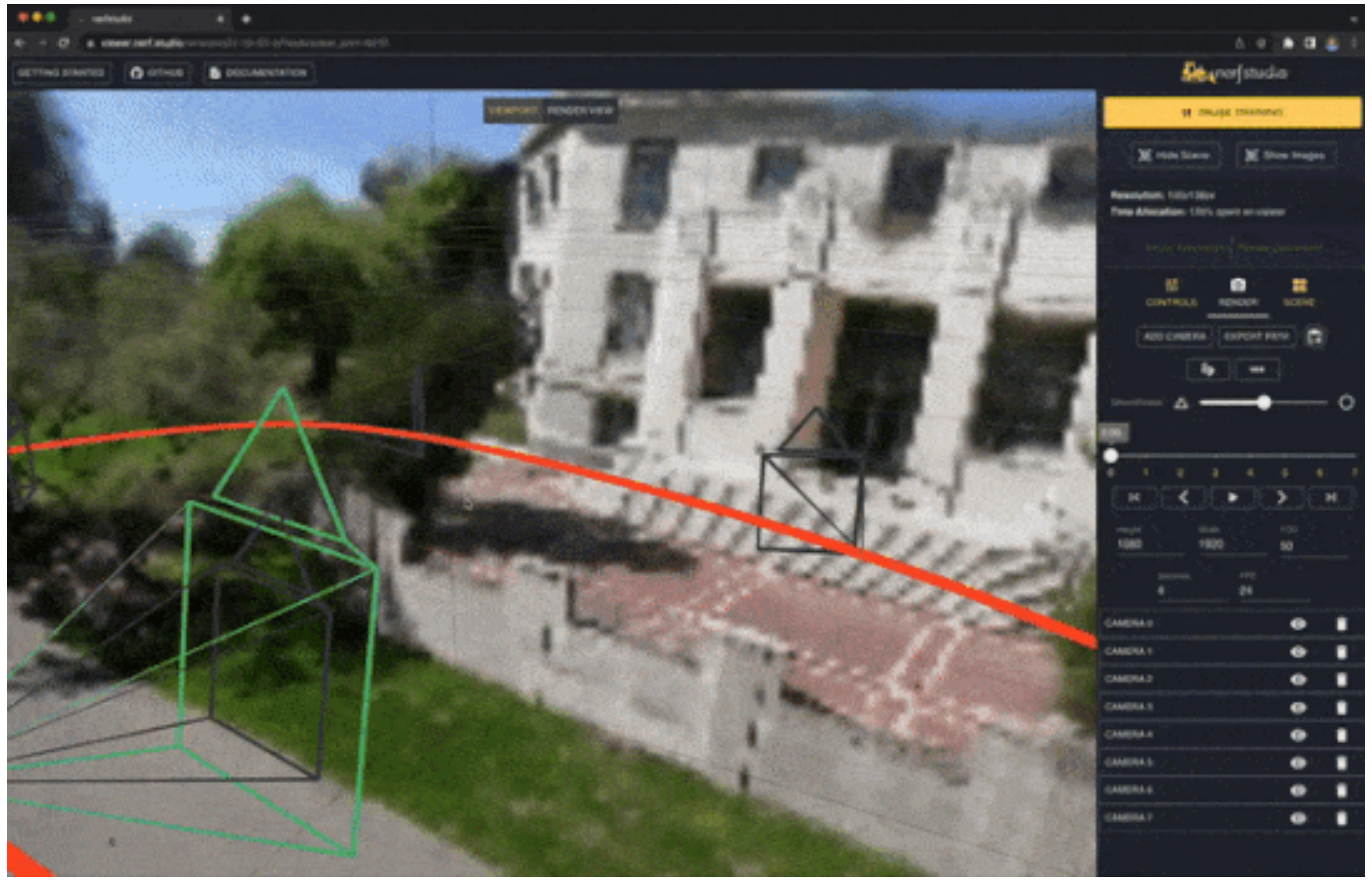
Következő Előadás:
Inverz Grafika a Gyakorlatban



3DGS/NeRF rekonstrukciós szoftverek. Dinamikus/4D Gaussian Splatting

Következő Előadás:

Inverz Grafika a Gyakorlatban



3DGS/NeRF rekonstrukciós szoftverek. Dinamikus/4D Gaussian Splatting