Polyhedral Complex Derivation from Piecewise Trilinear Networks

Jin-Hwa Kim
NAVER AI Lab & AI Institute of SNU
j1nhwa.kim@navercorp.com

github.com/naver-ai/tropical-nerf.pytorch
naver-ai.github.io/tropical-nerf

NAVER AI LAB

AIIS Artificial Intelligence Institute Seoul National University

Signed distance function (SDF)

- **Definition.** If Ω is a subset of a space X with a metric d, the SDF f is defined as:

$$f(x) = \begin{cases} -d(x, \partial\Omega) & \text{if } x \in \Omega \\ d(x, \partial\Omega) & \text{if } x \in \Omega^{\complement} \end{cases}$$

where $\partial\Omega$ denotes the boundary of Ω, and the metric with the boundary is:

$$d(x, \partial\Omega) := \inf_{y \in \partial\Omega} d(x, y),\ x^{\forall} \in X$$

- In the Euclidean space, d is the shortest distance from x to the boundary.

https://en.wikipedia.org/wiki/Signed_distance_function; A slightly different definition is used for consistency.

Properties in Euclidean space

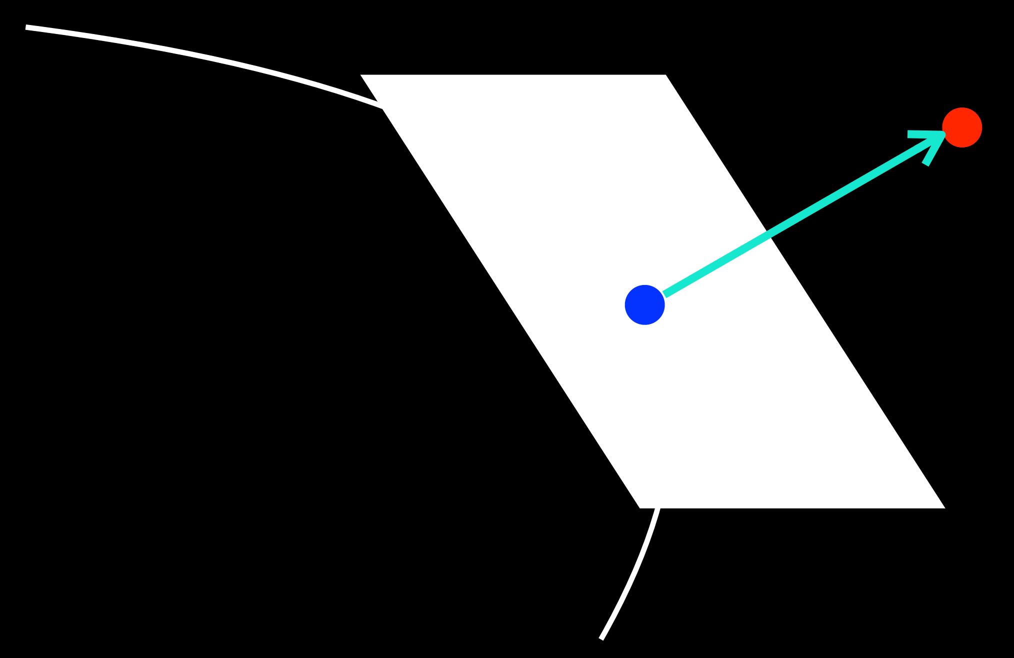
- For the Euclidean space with piecewise smooth boundary, the SDF is differentiable *almost everywhere*, and its gradient satisfies the *eikonal* equation:

$$|\nabla f| = 1$$

- Particularly, the gradient of f on the boundary of Ω is the *outward* normal vector:

$$\nabla f(x) = N(x).$$

- Therefore, the SDF is *a differentiable extension* of the normal vector field.



https://en.wikipedia.org/wiki/Signed_distance_function

Mesh from a SDF

- If an SDF is made up of ReLU-based neural networks, we can extract a mesh from the networks by leveraging *continuous piecewise affine (CPWA)* properties.
- We utilize the fact that 1) ReLU activation patterns create distinct linear regions, and 2) each neuron represents a folded hyperplane across these regions.

  👉 Formal descriptions using *tropical geometry* can be found in Sec. 3 & Appendix A.
- Since the number of linear regions *exponentially* grows with the depth of networks, Edge subdivision (Berzins, 2023) is an optimal algorithm that iterates over neurons, *not over the linear regions*, while subdividing the current set of edges.

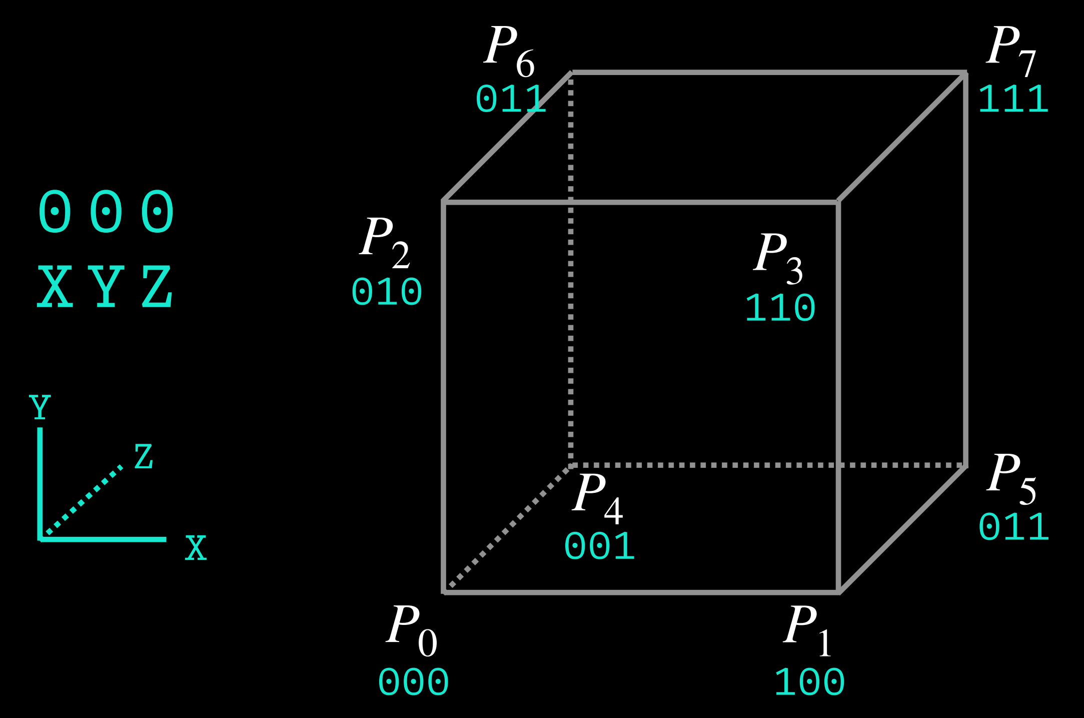
Motivation

- HashGrid (Müller et al., 2022) exploit trilinear interpolation to achieve fast convergence and mitigate spectral bias.
- Can we still analytically extract 3D mesh from the learned SDF?
  - 👉 *Eikonal constraint* makes the parameterized trilinear interpolation *continuous piecewise affine (CPWA)* function (Thm. 4.5 & Coro. 4.6).
  - 👉 Small high-resolution grids further reduce approximation errors since they can better fit curves with finer linear segments.
- For discussion, we define $\tau(\mathrm{x})$ as the HashGrid function with a single output.

Hypersurface and eikonal constraint

Theorem 4.5 (Hypersurface and eikonal constraint)**.** A hypersurface $\tau(\mathrm{x}) = 0$ passing two points $\tau(\mathrm{x}_0) = \tau(\mathrm{x}_7) = 0$ while $\tau(\mathrm{x}_{1\ldots6}) \neq 0$ for the remaining six points. These points form a cube, with x_0 and x_7 positioned on the diagonal of the cube. The hypersurface satisfies the eikonal constraint $\|\nabla\tau(\mathrm{x})\|_2^2 = 1$ for all $\mathrm{x} \in [0,\ 1]^3$. Then, the hypersurface of $\tau(\mathrm{x}) = 0$ is a plane.

Proof sketch. The eikonal constraint makes the surfaces of an SDF smooth and coherent structure. The linearity would make planar surfaces. For the proof, we calculate the *second derivatives of trilinear interpolation to be zeros* and find the constraints to satisfy the linearity.



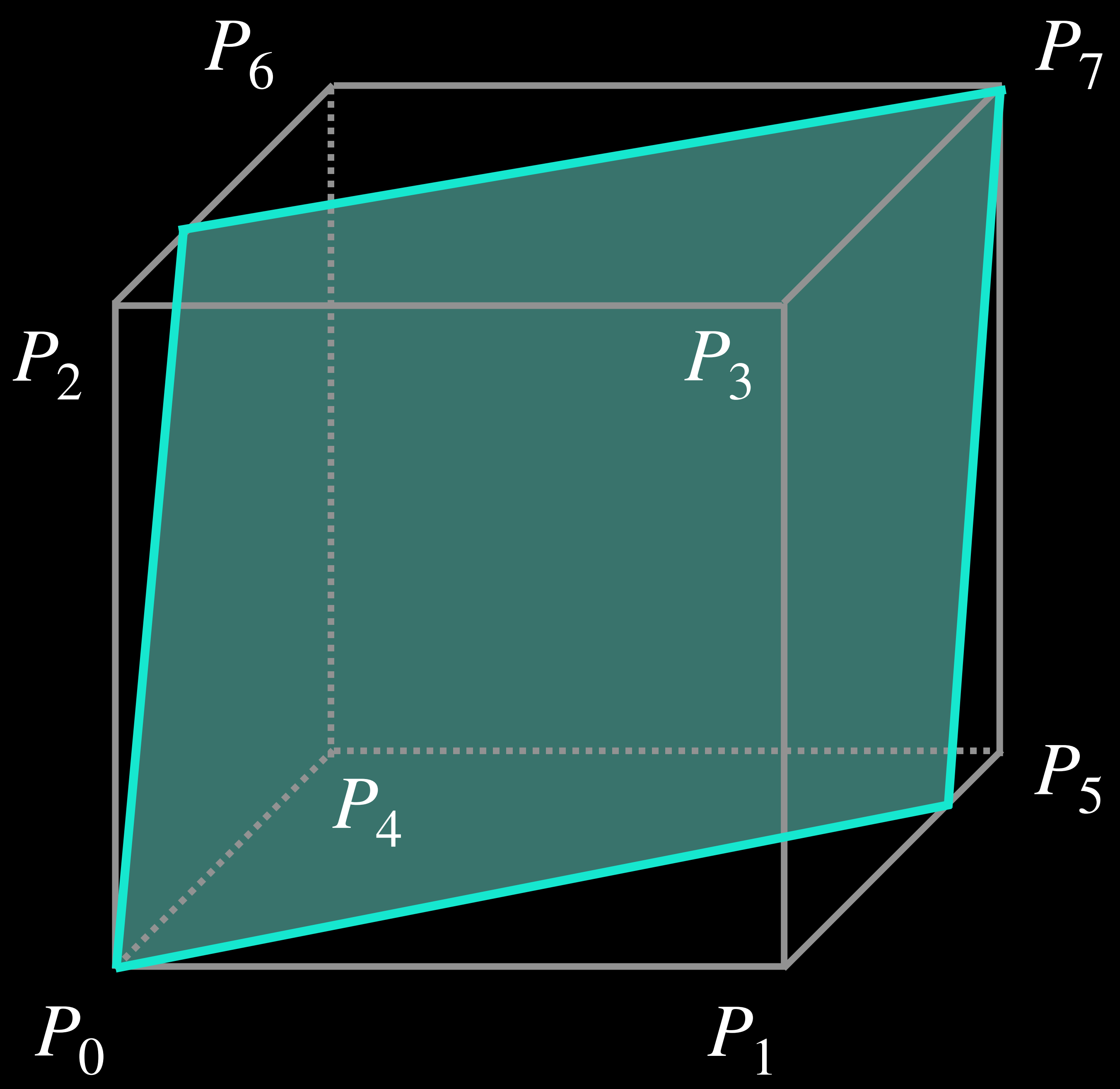
Implications of Thm. 4.5

- To satisfy the eikonal constraint, the following equations are true:

$$\left.\begin{aligned} \tau(\mathrm{x}_1) + \tau(\mathrm{x}_6) &= 0 \\ \tau(\mathrm{x}_2) + \tau(\mathrm{x}_5) &= 0 \\ \tau(\mathrm{x}_3) + \tau(\mathrm{x}_4) &= 0 \end{aligned}\right\} \textit{opposite two vertices}$$

$$\left.\begin{aligned} \tau(\mathrm{x}_1) + \tau(\mathrm{x}_2) + \tau(\mathrm{x}_4) &= 0 \\ \tau(\mathrm{x}_3) + \tau(\mathrm{x}_5) + \tau(\mathrm{x}_6) &= 0 \end{aligned}\right\} \textit{two diagonal groups}$$

where the hash table entries are $P_i = \tau(\mathrm{x}_i)$.



Ref. Appendix D Theoretical proofs

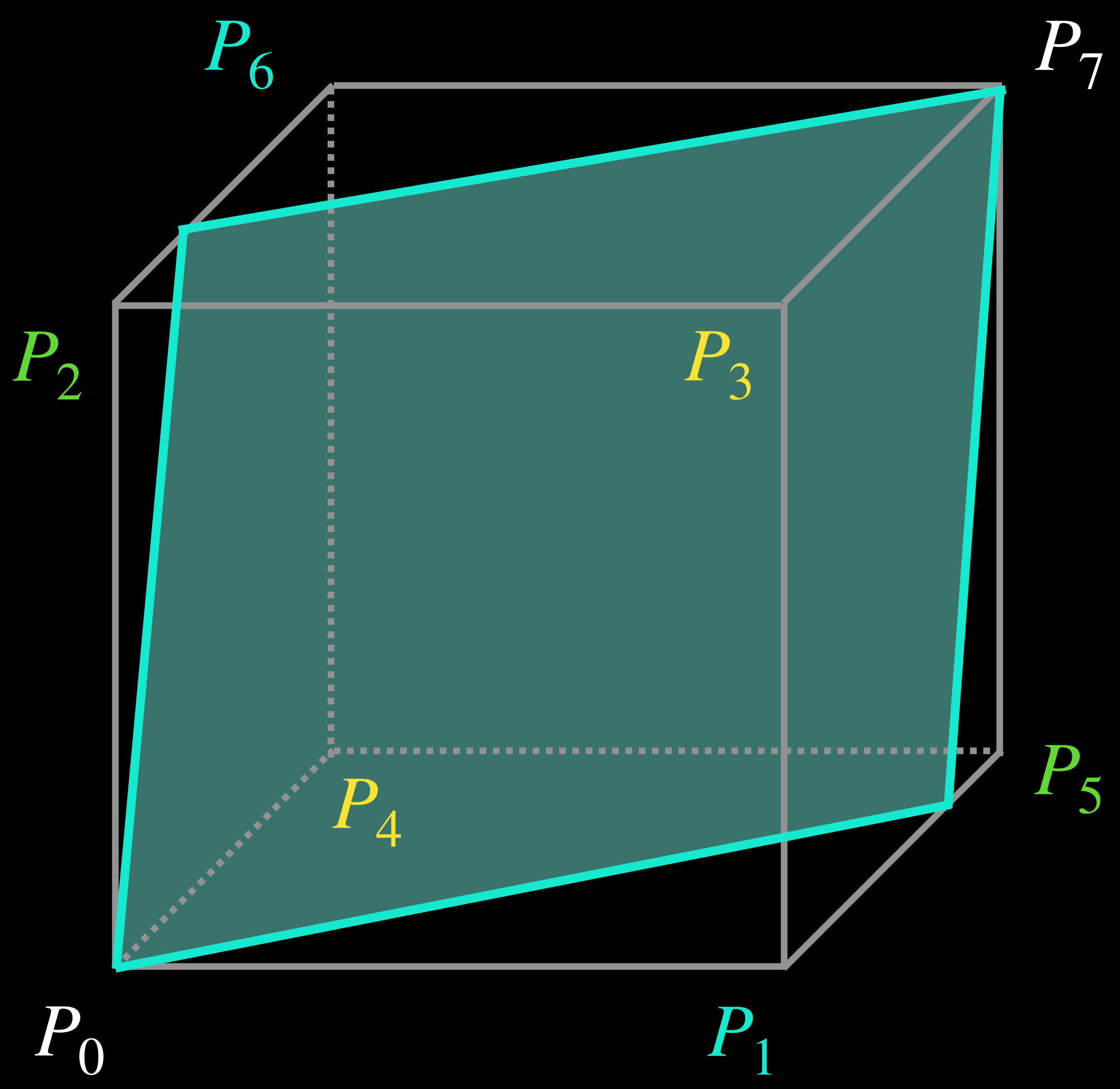
Implications of Thm. 4.5

- To satisfy the eikonal constraint, the following equations are true:

$$\left.\begin{aligned} \tau(\mathrm{x}_1) + \tau(\mathrm{x}_6) &= 0 \\ \tau(\mathrm{x}_2) + \tau(\mathrm{x}_5) &= 0 \\ \tau(\mathrm{x}_3) + \tau(\mathrm{x}_4) &= 0 \end{aligned}\right\} \textit{opposite two vertices}$$

$$\left.\begin{aligned} \tau(\mathrm{x}_1) + \tau(\mathrm{x}_2) + \tau(\mathrm{x}_4) &= 0 \\ \tau(\mathrm{x}_3) + \tau(\mathrm{x}_5) + \tau(\mathrm{x}_6) &= 0 \end{aligned}\right\} \textit{two diagonal groups}$$

where the hash table entries are $P_i = \tau(\mathrm{x}_i)$.



Ref. Appendix D Theoretical proofs

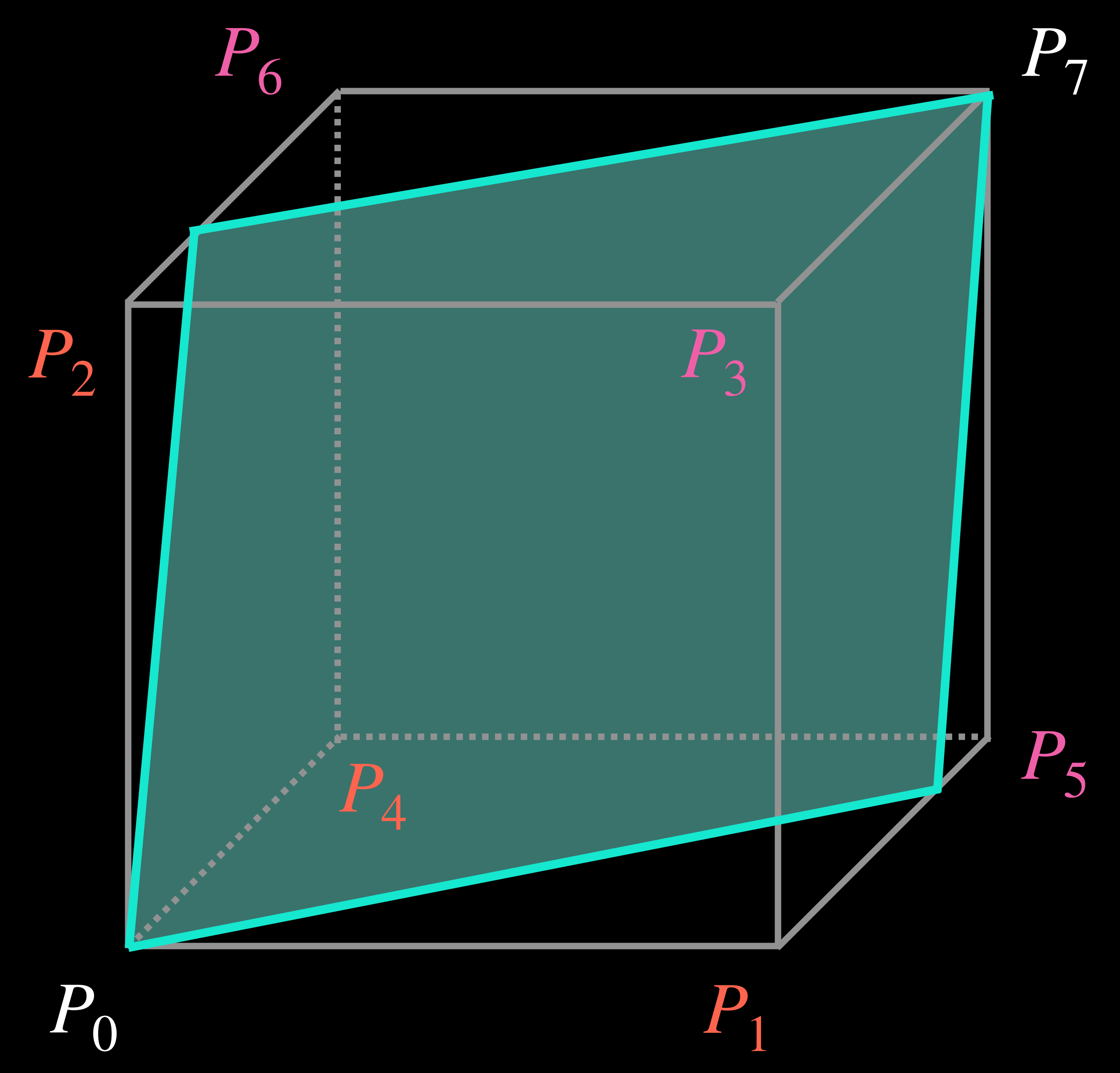
Implications of Thm. 4.5

- To satisfy the eikonal constraint, the following equations are true:

$$\left.\begin{aligned} \tau(x_1) + \tau(x_6) &= 0 \\ \tau(x_2) + \tau(x_5) &= 0 \\ \tau(x_3) + \tau(x_4) &= 0 \end{aligned}\right\} \textit{opposite two vertices}$$

$$\left.\begin{aligned} \tau(x_1) + \tau(x_2) + \tau(x_4) &= 0 \\ \tau(x_3) + \tau(x_5) + \tau(x_6) &= 0 \end{aligned}\right\} \textit{two diagonal groups}$$

where the hash table entries are $P_i = \tau(x_i)$.



Ref. Appendix D Theoretical proofs

Flatness error

- To satisfy the eikonal constraint, the following equations are true:

$$\left.\begin{aligned} &\text{●}\ \ \tau(\mathrm{x}_1) + \tau(\mathrm{x}_6) = 0 \\ &\text{▲}\ \ \tau(\mathrm{x}_2) + \tau(\mathrm{x}_5) = 0 \\ &\text{■}\ \ \tau(\mathrm{x}_3) + \tau(\mathrm{x}_4) = 0 \end{aligned}\right\}\ \textit{opposite two vertices}$$

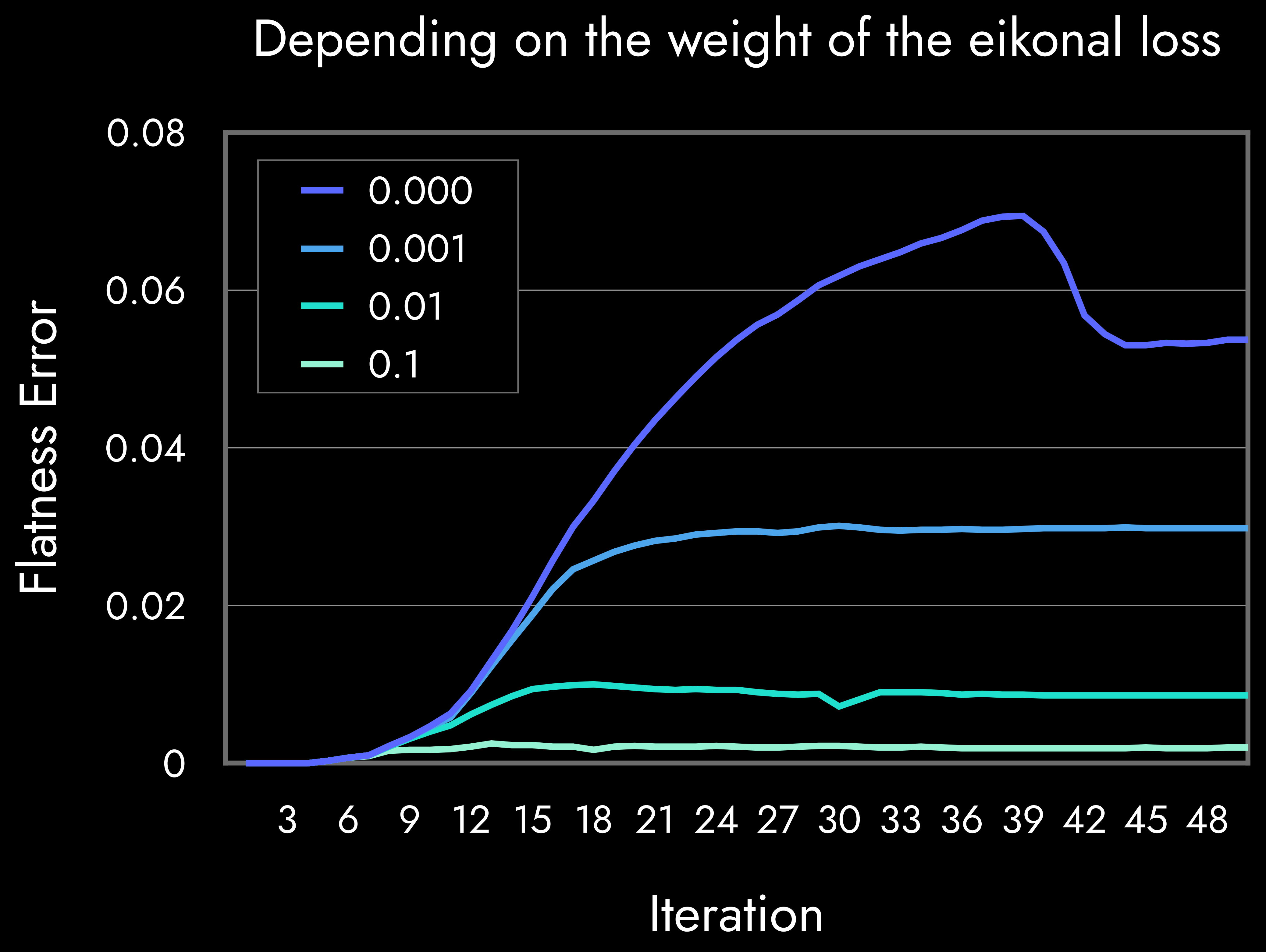
$$\left.\begin{aligned} &\text{◆}\ \ \tau(\mathrm{x}_1) + \tau(\mathrm{x}_2) + \tau(\mathrm{x}_4) = 0 \\ &\text{★}\ \ \tau(\mathrm{x}_3) + \tau(\mathrm{x}_5) + \tau(\mathrm{x}_6) = 0 \end{aligned}\right\}\ \textit{two diagonal groups}$$

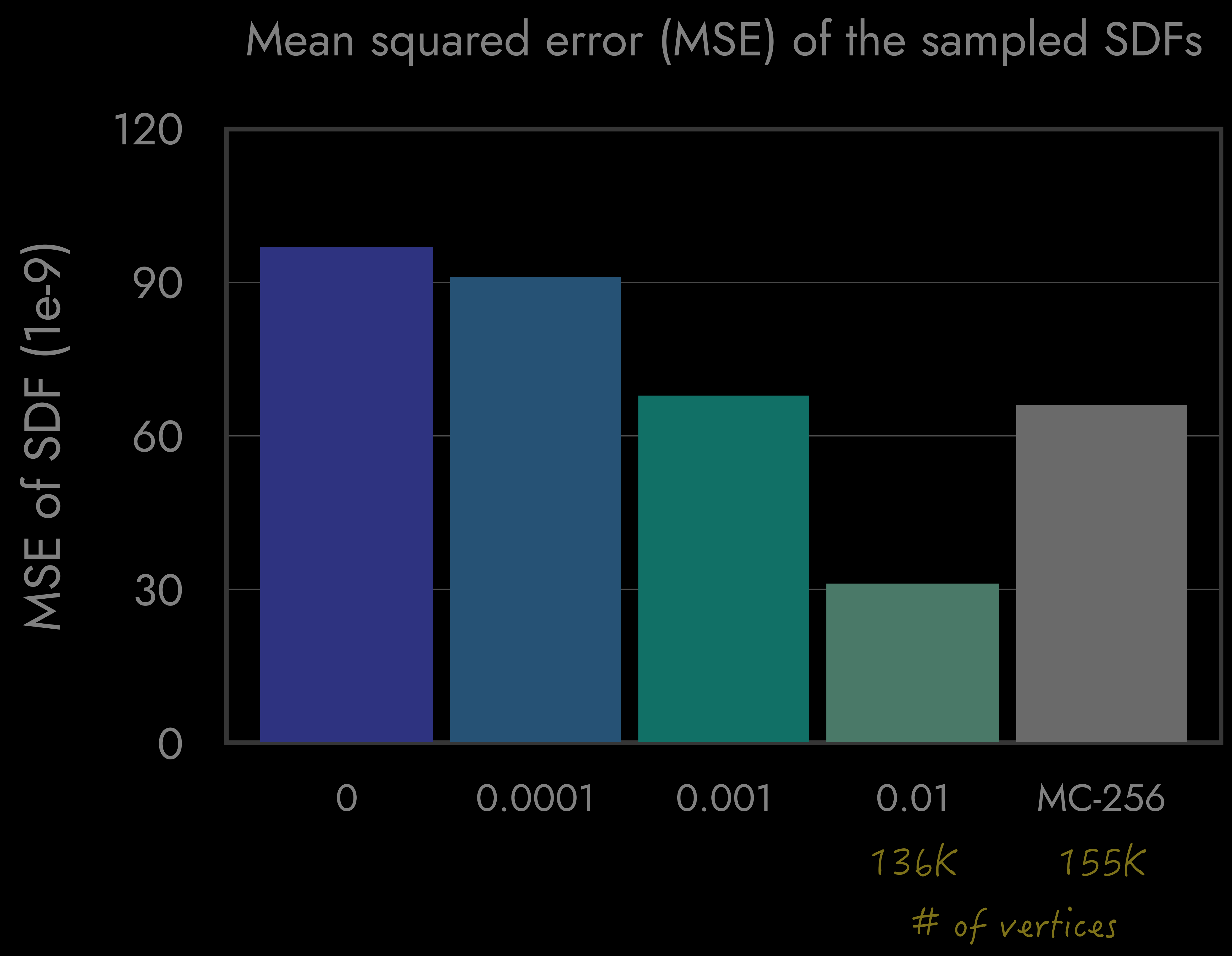
- The mean absolute error (MAE) of *flatness* is defined as:

$$\mathbb{E}_{\mathscr{E}}\left[\frac{1}{6}\left(\lVert \text{●} \rVert_1 + \lVert \text{▲} \rVert_1 + \lVert \text{■} \rVert_1\right) + \frac{1}{4}\left(\lVert \text{◆} \rVert_1 + \lVert \text{★} \rVert_1\right)\right].$$

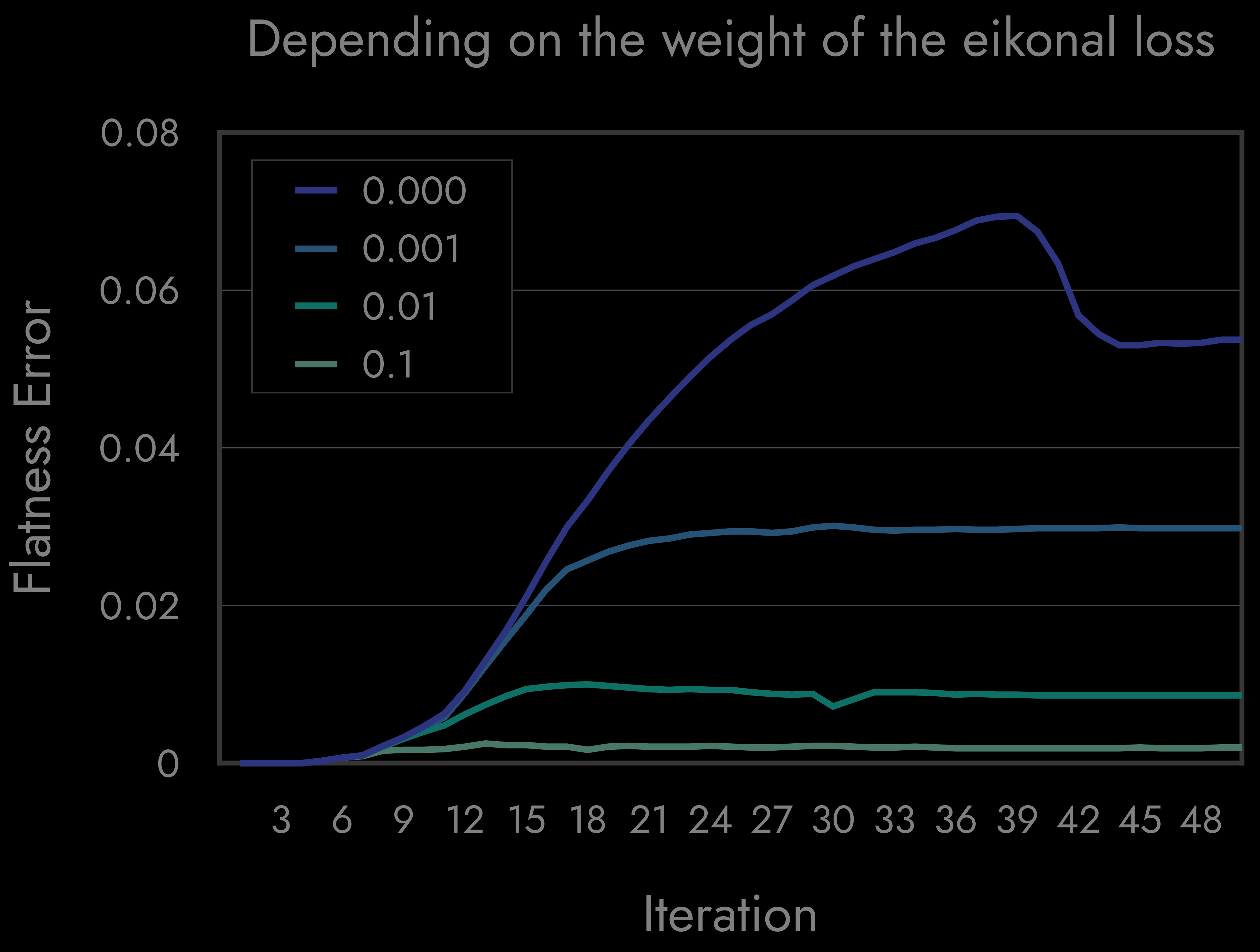
Ref. Appendix D Theoretical proofs

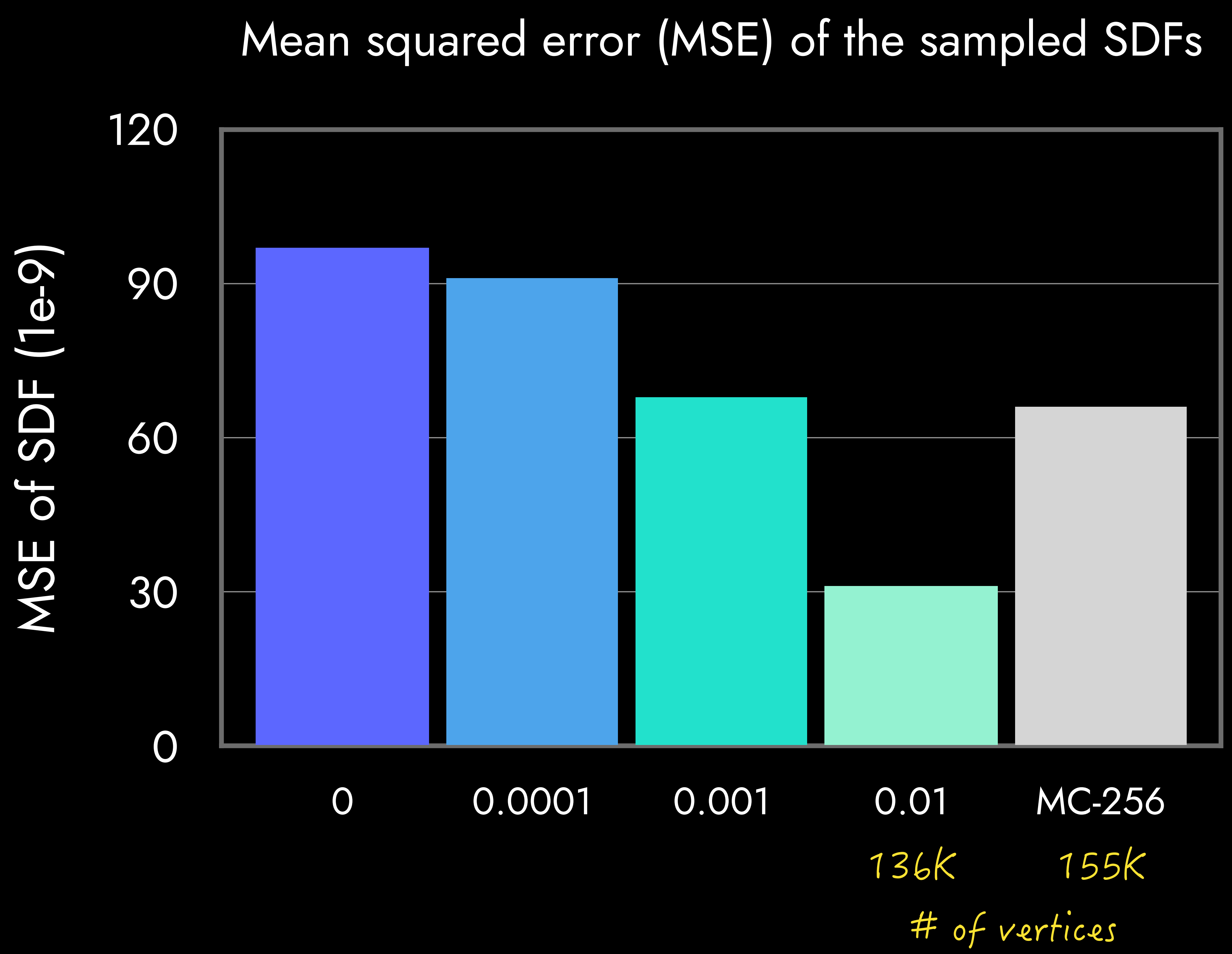
Empirical validations





Empirical validations

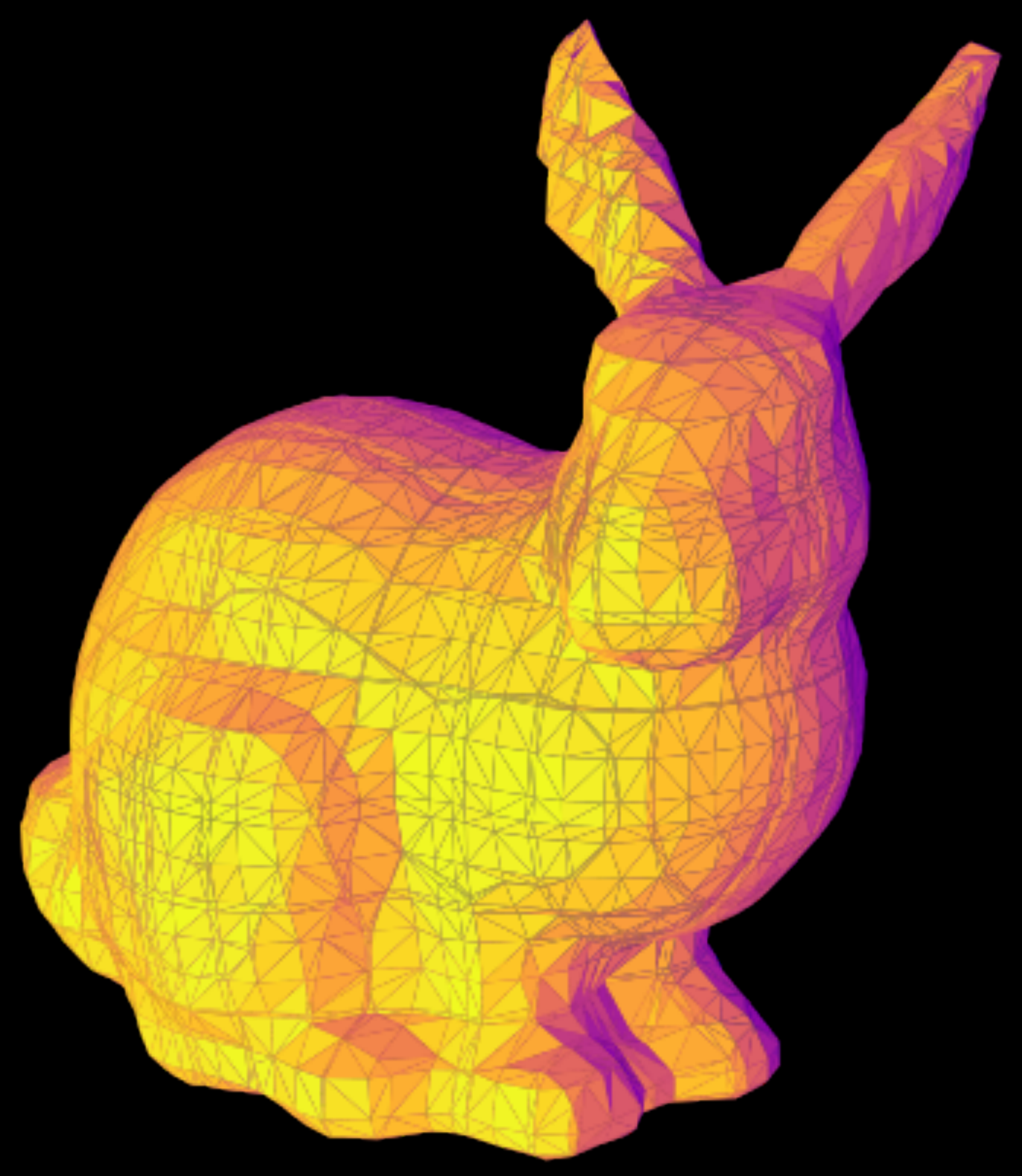




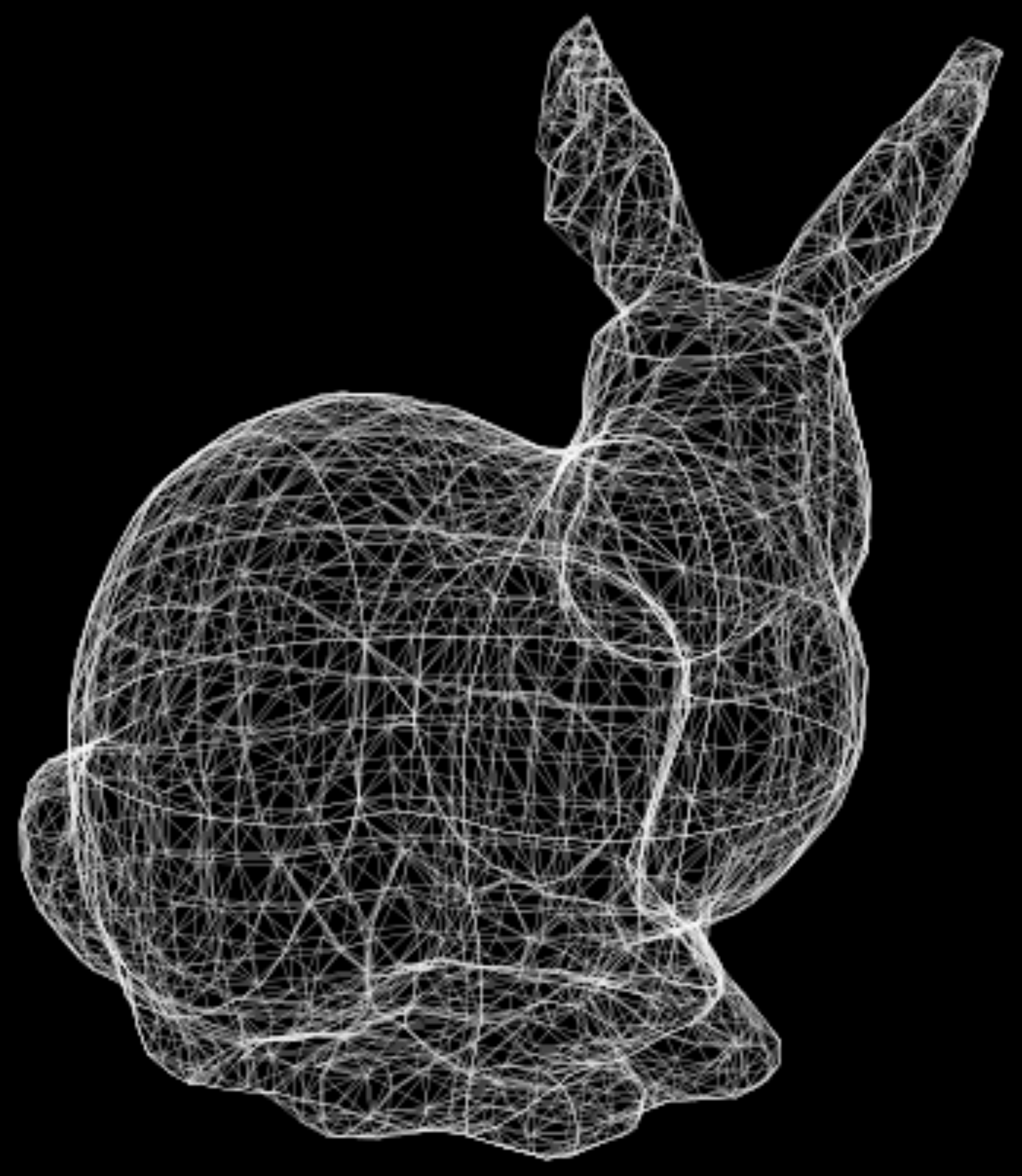
Discussions

- *Fast convergence*. Hash table entries are easy to optimize as learnable parameters.
- *Selective learning*. Eikonal constraint applies mainly near surfaces, focusing on a small subset of space.
- *Online adaptivity*. Allocations of hash table entries concentrate on a small region, and finer grids will experience fewer collisions while hashing (Müller et al., 2022).

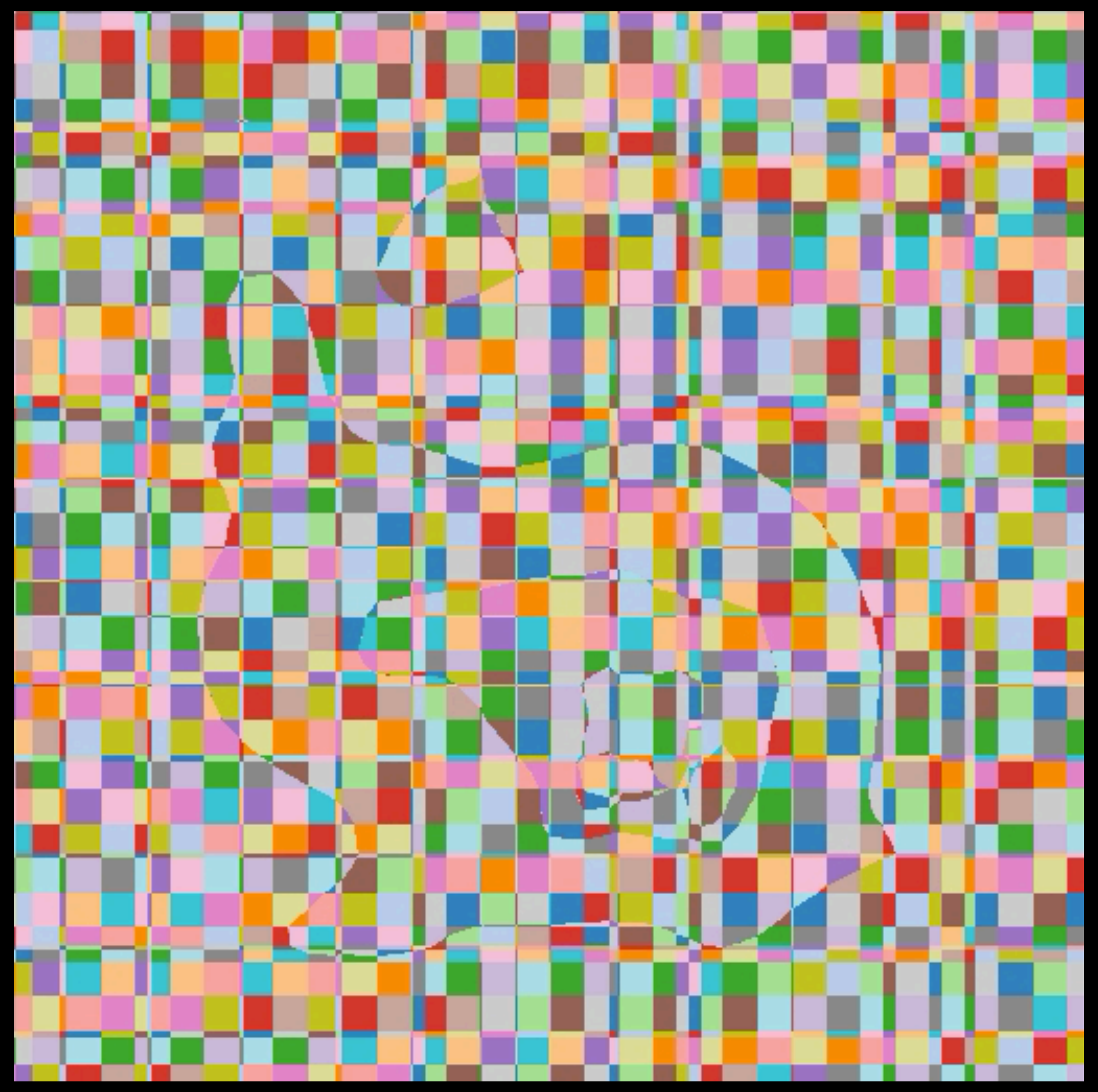
Visualizations



Normal map

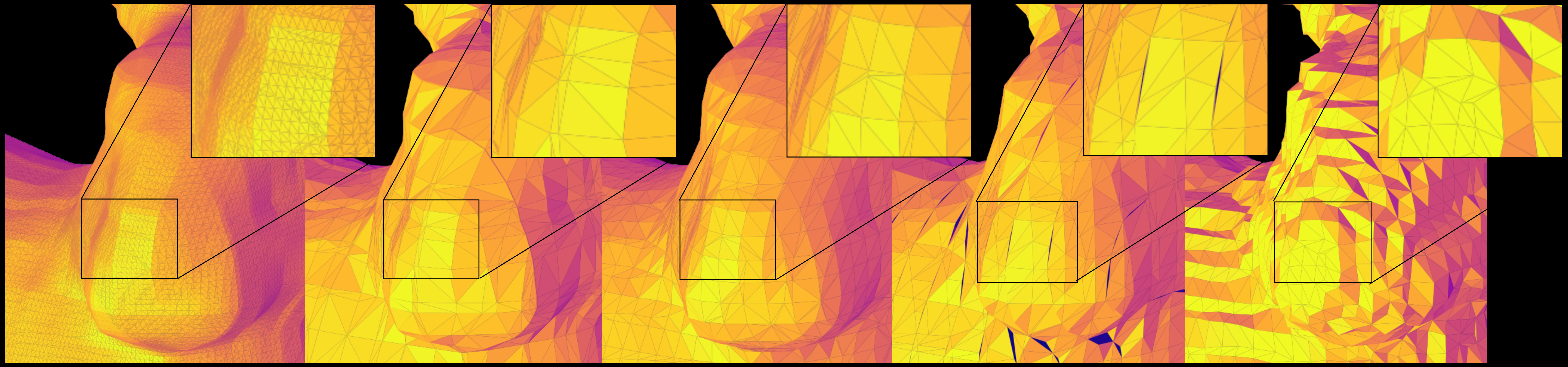


Skeleton



Trilinear regions shifting along the z-axis

Nose-to-nose comparison



MC 256
(92K Vertices as GT)

Ours
(4.5K Vertices)

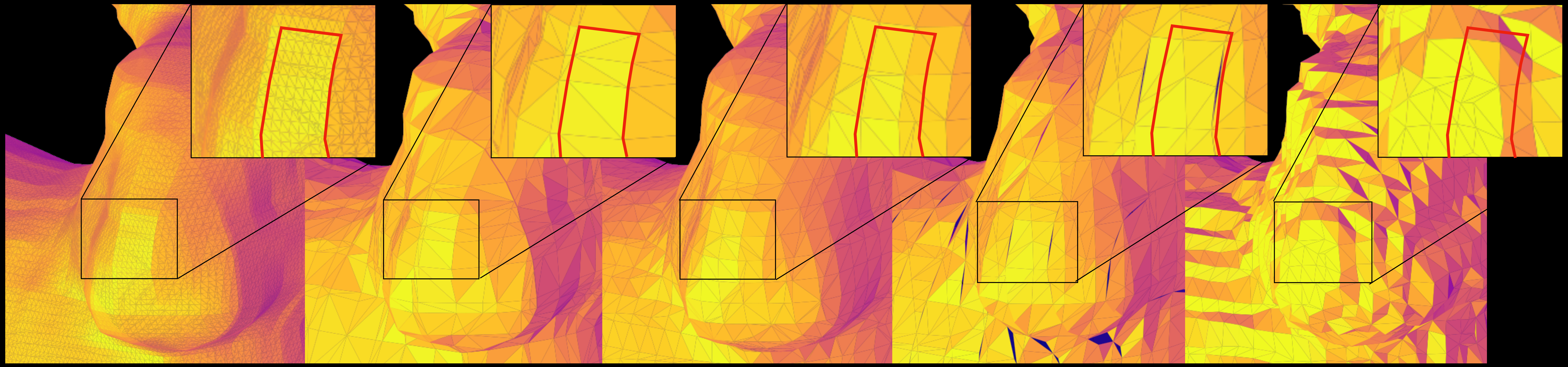
MC 64
(5.6K Vertices)

MT 32
(4.7K Vertices)

NDC 64
(5.6K Vertices)

MT (Marching Tetrahedra), NDC (Neural Dual Contour; Chen et al., 2022)

Nose-to-nose comparison



MC 256
(92K Vertices as GT)

Ours
(4.5K Vertices)

MC 64
(5.6K Vertices)

MT 32
(4.7K Vertices)

NDC 64
(5.6K Vertices)

MT (Marching Tetrahedra), NDC (Neural Dual Contour; Chen et al., 2022)

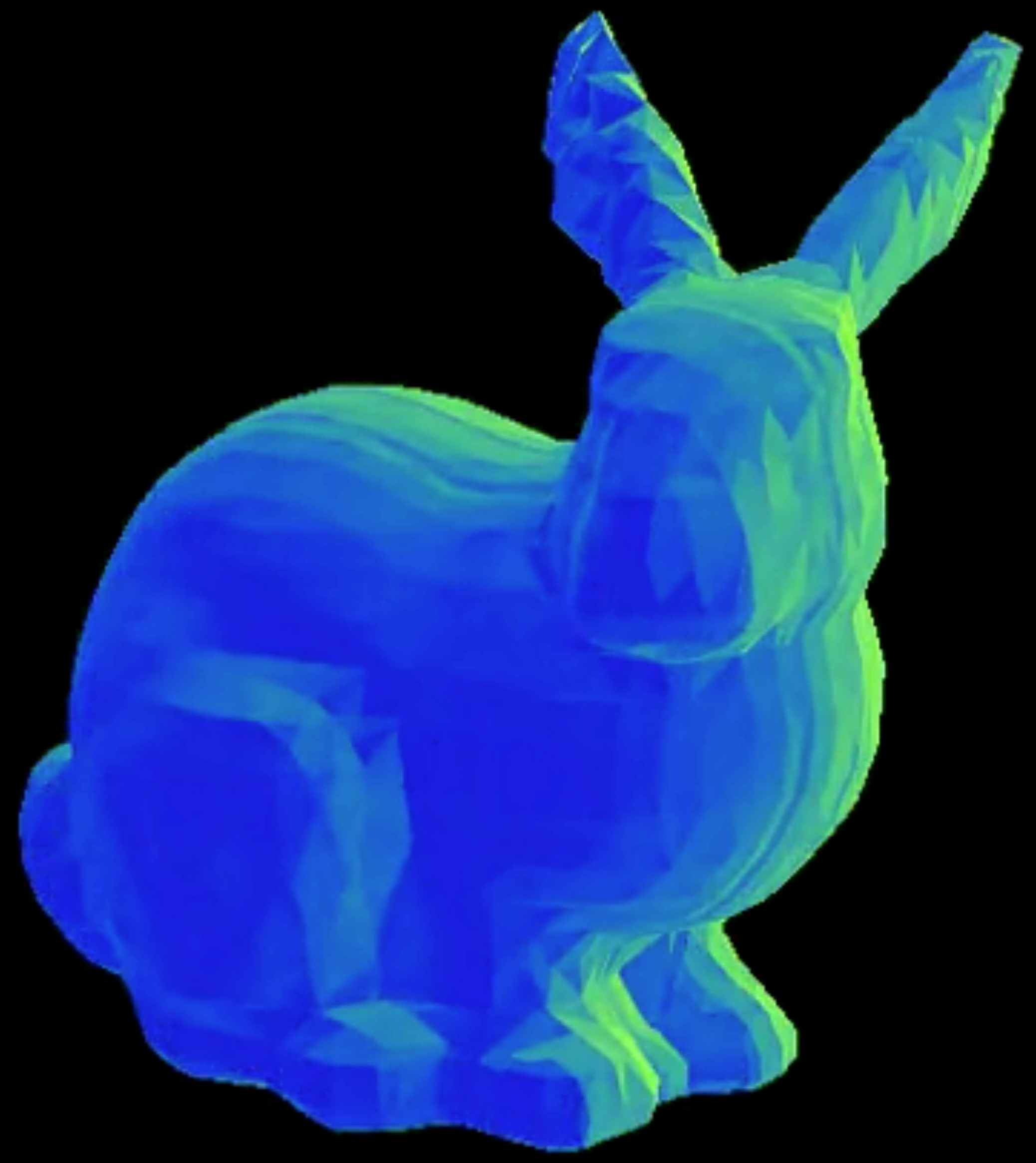
Deep dive into our paper

- Algorithms describing the whole procedure (Alg. 1 & 2)
- An approximation to get the intersection of three curved hypersurfaces (Thm. 4.7)
- Quantitative results on the Stanford 3D Scanning repository (Curless & Levoy, 1996) report the Chamfer distance and efficiency, the angular distance, and the time spent.
- The publicly available code[1] allows you to review implementation details for batch computations optimized for maximum parallelization.

[1] https://github.com/naver-ai/tropical-nerf.pytorch

Conclusions

- We present novel theoretical insights and a practical methodology for precise mesh extraction, employing *piecewise trilinear networks*.
- This provides *a theoretical exposition of the eikonal constraint*, revealing that within the trilinear region, the hypersurface transforms into a plane.
- We hope this novel discovery will inspire future work that explores innovative applications and further advancements in the field.



Thank you all!

I want to express my sincere appreciation to my brilliant colleagues, Sangdoo Yun, Dongyoon Han, and Injae Kim, for their contributions to this work. Their constructive feedback and guidance have been instrumental in shaping the work. I also express my sincere thanks to the anonymous reviewers for their help in improving the manuscript. The NAVER Smart Machine Learning (NSML) platform (Kim et al., 2018) was used for experiments.