


Sculpting procedural noise for real-time visualization of astronomical objects

M2 MoSIG Research internship - 2023

Mathéo MOINET
Supervisor : Fabrice NEYRET

Context: Galaxy walkthrough

Continuation of veRTIGE: a virtual tour through our galaxy

- Real-time walkthrough through the galaxy
- Hubble-like quality

Usage in digital planetariums - RSA cosmos

Internship focus:

- Interstellar dust clouds (nebula, galaxy arm, solar flare, ...)

How to model these objects ?

No 3D groundtruth:

- 2D view from Earth
- Sparse physical knowledge

Case study: Astronomical objects

No 3D groundtruth:

- 2D view from Earth
- Sparse physical knowledge

A few observable properties:

1. Volumetric & semi-transparent

Case study: Astronomical objects

No 3D groundtruth:

- 2D view from Earth
- Sparse physical knowledge

A few observable properties:

1. Volumetric & semi-transparent
2. Large voids

Case study: Astronomical objects

No 3D groundtruth:

- 2D view from Earth
- Sparse physical knowledge

A few observable properties:

1. Volumetric & semi-transparent
2. Large voids
3. Clusters of high density with a macro shape (tube, spiral)

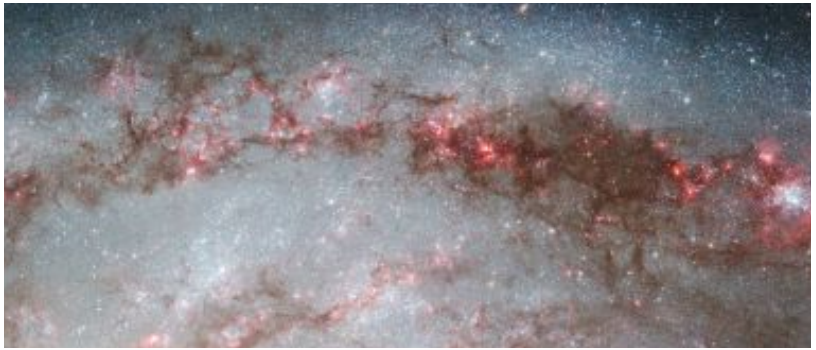
Case study: Astronomical objects

No 3D groundtruth:

- 2D view from Earth
- Sparse physical knowledge

A few observable properties:

1. Volumetric & semi-transparent
2. Large voids
3. Clusters of high density with a macro shape (tube, spiral)
4. Stochastic details



Case study: Astronomical objects

No 3D groundtruth:

- 2D view from Earth
- Sparse physical knowledge

A few observable properties:

1. Volumetric & semi-transparent
2. Large voids
3. Clusters of high density with a macro shape (tube, spiral)
4. Stochastic details
5. Anisotropy caused by deformation

Goals & a priori knowledge

Data structures and **algorithms** allowing:

- **Real-time volumetric** rendering (60 fps)
- **High details** resolution (but fits in GPU memory, **4-16 GB**)
- Ease of design
- ~~Infinite scene~~

A priori knowledge:

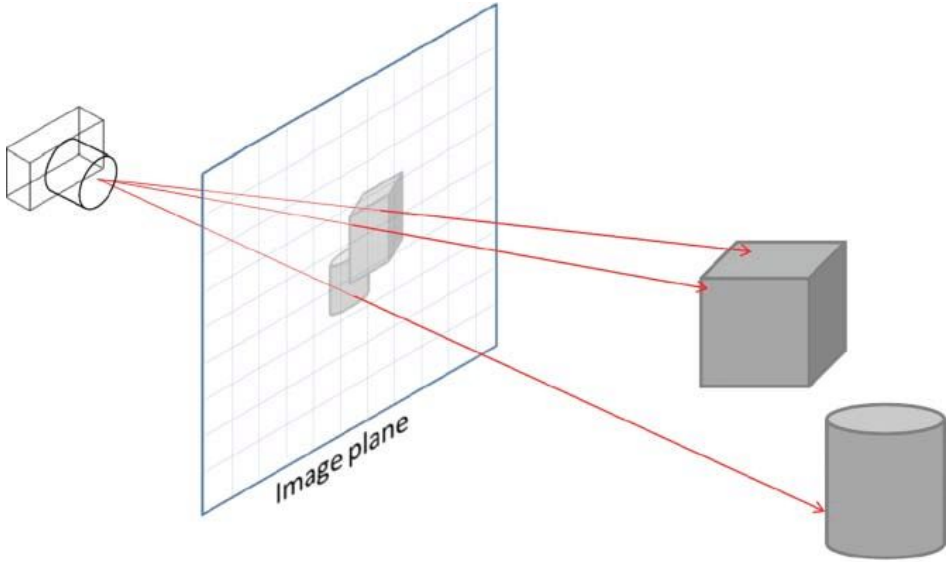
- Big empty spaces
- Medium scale: **cloud clusters** with **simple shape**
- Small scale: **anisotropic** “galactic dust cloud” **material**

2. State of the art

Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy

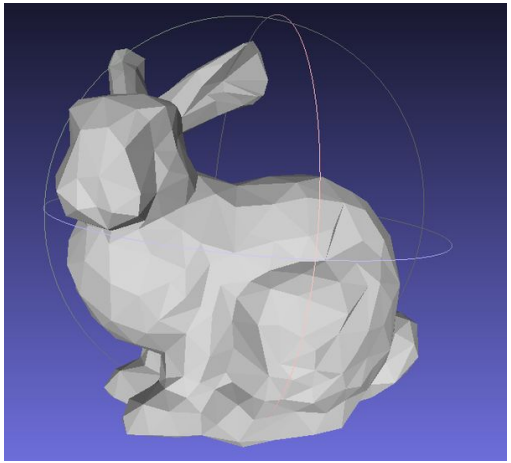
2.1. How to represent & render scenes ?

Surface ray casting:



For each **ray**:

- Compute **intersection** with **every object**
- Find **closest object**
- Compute **appearance** of object at intersection point



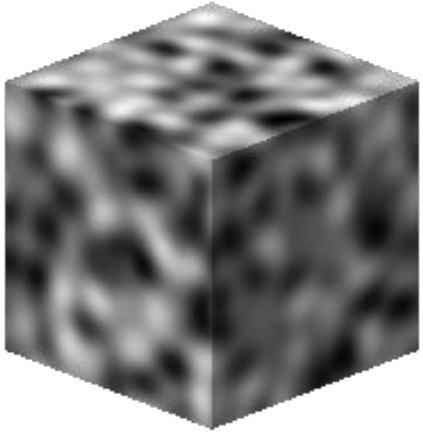
2.1. How to represent & render scenes ?

What if **no real “surface”** ?

2.1. How to represent & render scenes ?

Rendering heterogeneous participating medium
➜ **integrate density along ray**

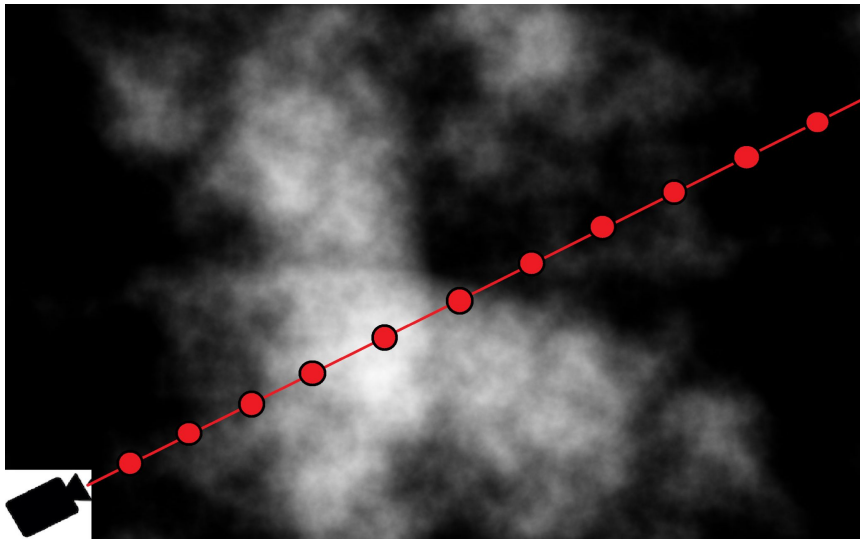
Density field:



Describe **density** at **every position** ***P*** = (x,y,z) in the scene.

$$density(P) : \mathbb{R}^3 \rightarrow \mathbb{R}$$

Volumetric ray marching:
numerical integration by **sampling**



2. State of the art - comparison

1.2. Plain RGBA volumes
1.3. Fully procedural volumes
1.4. Mesh-based volumes
1.5. Implicit volumes

Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy
Plain volume					
Fully procedural					
Mesh based					
Implicit volume					

2.2. Plain RGBA volumes (ex: Ilumbra^{TM})

Orion nebula 3D image (512^3):

- Memory / quality tradeoff ($1024^3 \simeq$ 4 GB)
- Requires skilled astronomical experts and artists
- Slow to render

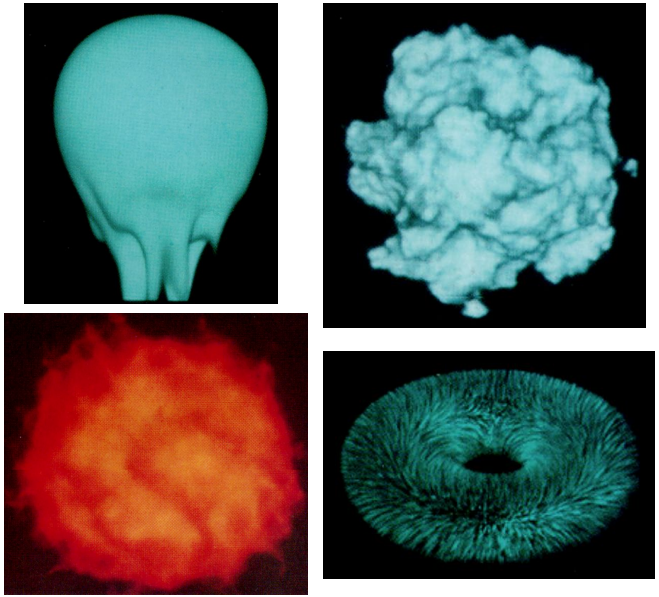
2.2. State of the art - comparison

Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy
Plain volume	- -	No	- -	-	- -
Fully procedural					
Mesh based					
Implicit volume					

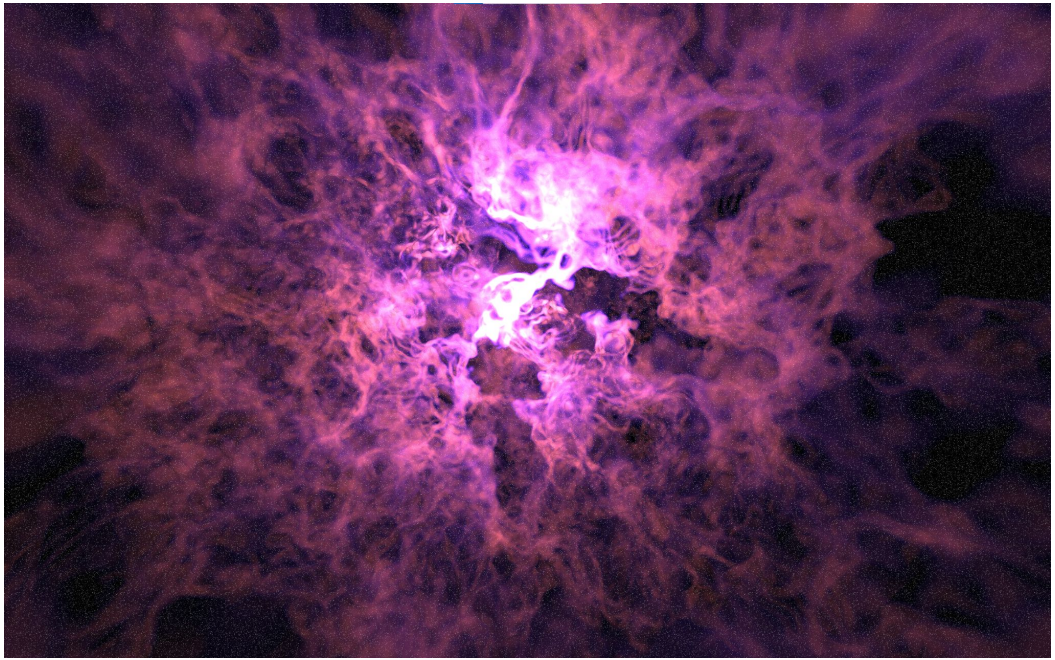
2.3. Procedural volumes

$$noise(P) : \mathbb{R}^3 \rightarrow \mathbb{R}$$
$$density(P) = noise(P)$$

“Hypertextures” [Perlin, 89]



Procedural nebula (Erwan Leria):



2.3. State of the art - comparison

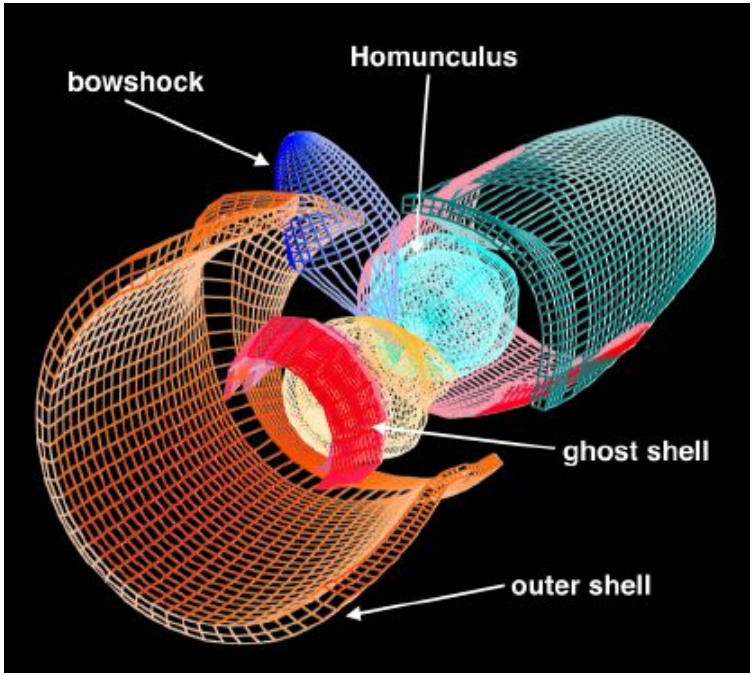
Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy
Plain volume	- -	No	- -	-	- -
Fully procedural	0	Yes	- -	- -	-
Mesh based					
Implicit volume					

2.4. Mesh-based volume (ex: Shape™)

Orion nebula (Shape™):

Homunculus Nebula (Shape™):

[Mehner, et al., 2016]



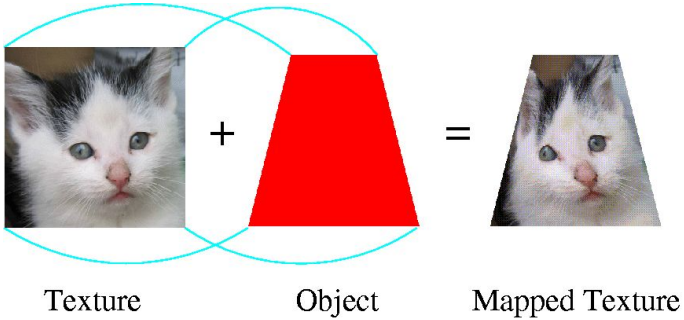
Texture mapping

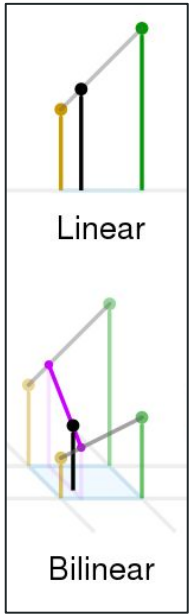
$$map(P) : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$
$$density(P) = noise(map(P))$$

$noise(P)$: **Procedural material**

$map(P)$: **Material space coordinates**

Isotropic material + **anisotropic** mapping
➔ **anisotropic material**





With a mesh :

$map(P)$: computed by **linear interpolation** of map values **stored at mesh nodes**

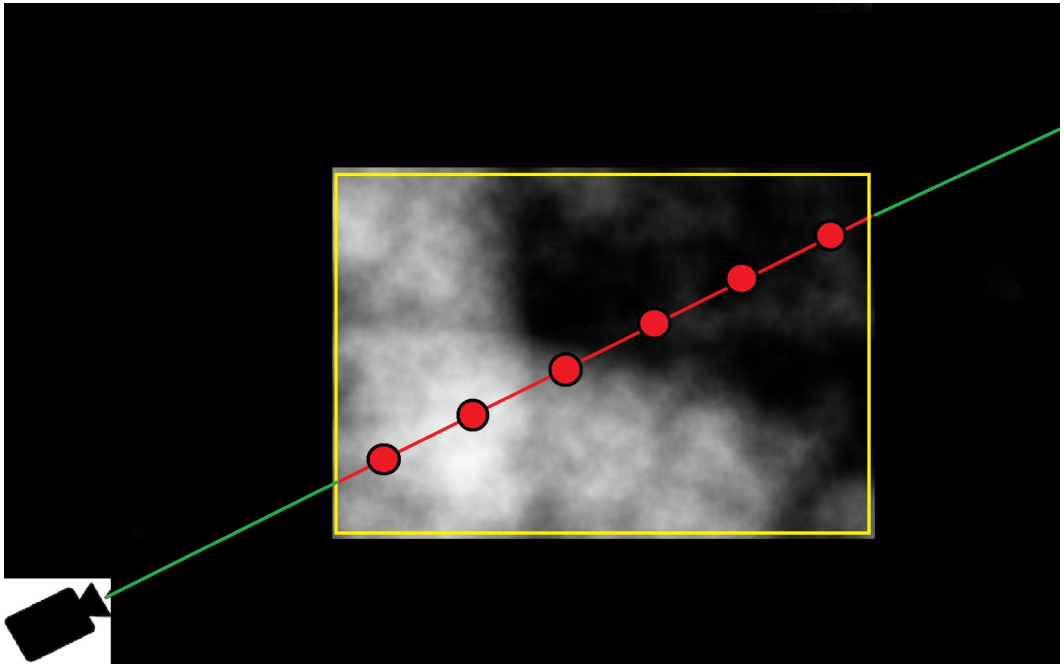
Linear interpolation:

From **2 samples**, reconstruct **linear approximation** of the function in the interval of the samples.

Intuitively: **weighted mean** between samples

2D extension : **Bilinear** interpolation
3D extension : **Trilinear** interpolation

Empty space skipping

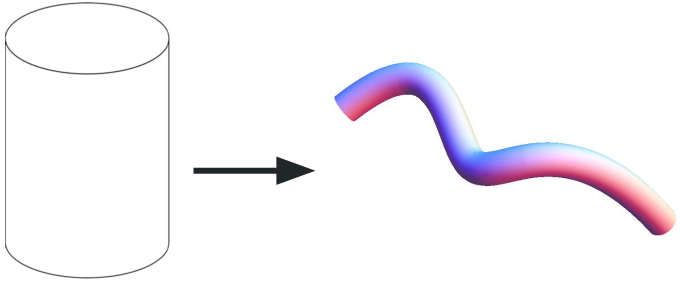


2.4. State of the art - comparison

Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy
Plain volume	- -	No	- -	-	- -
Fully procedural	0	Yes	- -	- -	-
Mesh based	-	Yes	+	+	+
Implicit volume					

2.5. Implicit volume (ex: [MM22])

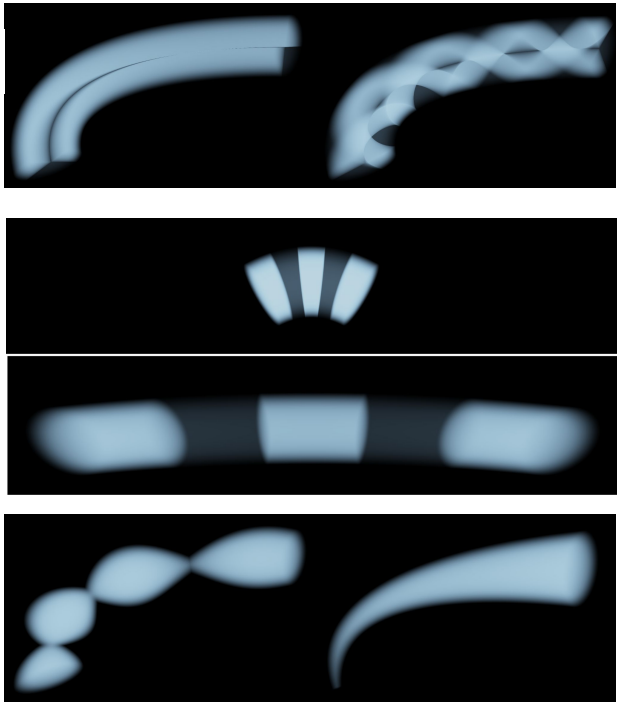
Spline tubes - Generalized cylinders:

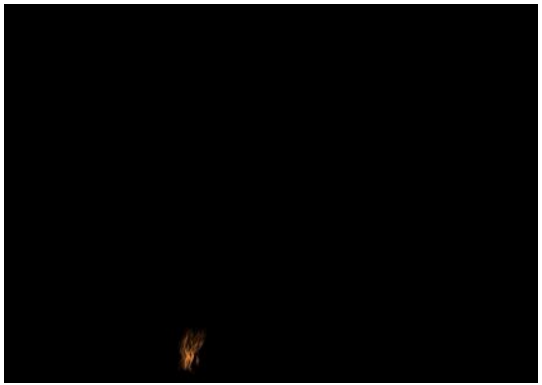


$$density(P) = noise(map(P))$$

$map(P)$ computed directly on the fly

Torsion, stretch, varying radius

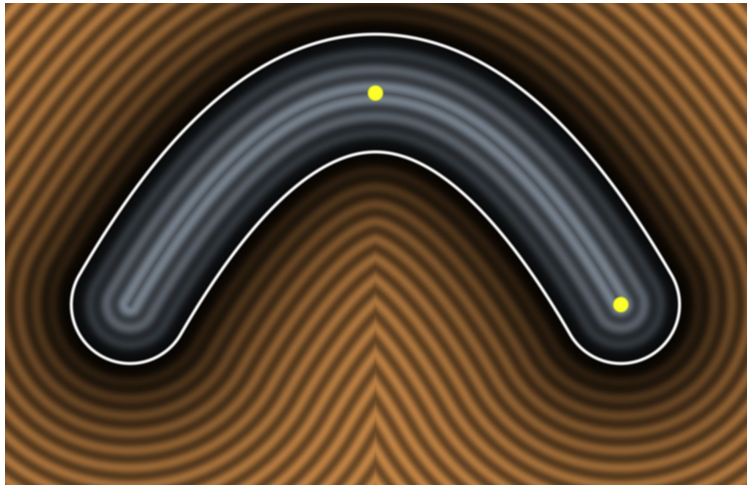




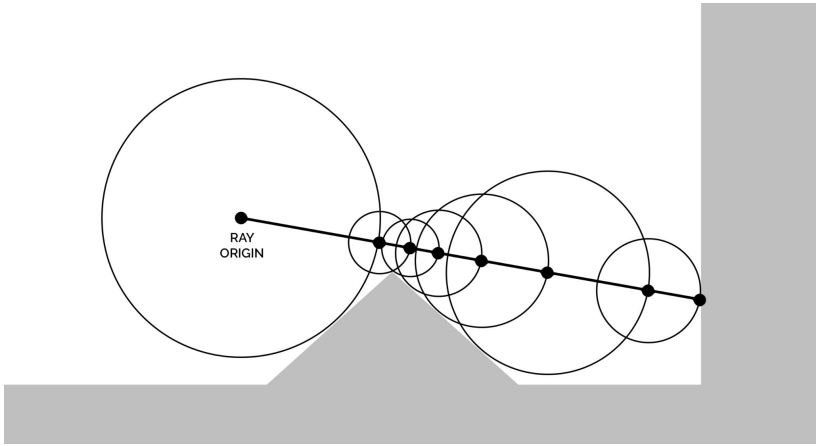
Signed Distance Field (SDF)

$$SDF(P) : \mathbb{R}^3 \rightarrow \mathbb{R}$$

$$SDF(P) = \begin{cases} distance(P), & \text{if } P \text{ outside of the object} \\ -distance(P), & \text{if } P \text{ inside of the object} \\ 0, & \text{if } P \text{ on the object} \end{cases}$$

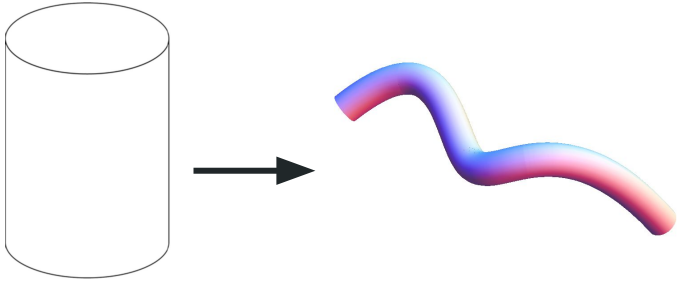


Sphere tracing [Hart, J. C. (1996)]:



2.5. Implicit volume (ex: [MM22])

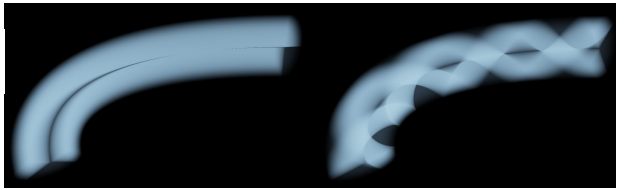
Spline tubes - Generalized cylinders:

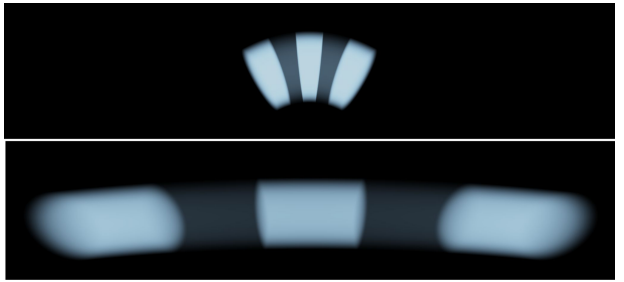


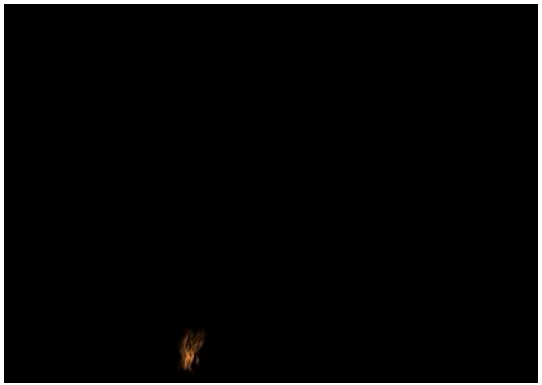
$$density(P) = noise(map(P))$$

$map(P)$ computed directly on the fly

Torsion, stretch, varying radius







2.5. State of the art - comparison

Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy
Plain volume	- -	No	- -	-	- -
Fully procedural	0	Yes	- -	- -	-
Mesh based	-	Yes	+	+	+
Implicit volume	0	Yes	+	+	+

State of the art: conclusions

Method	Memory	Infinite details	Ease to design	Render time	Local anisotropy
Plain volume	- -	No	- -	-	- -
Fully procedural	0	Yes	- -	- -	-
Mesh based	-	Yes	+	+	+
Implicit volume	0	Yes	+	+	+

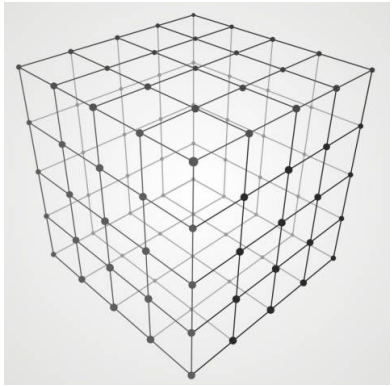
Our takeaway:

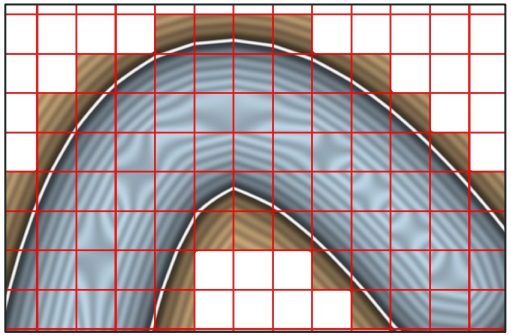
- Separate shape and material: □
- Skip empty spaces: □
- Procedural material: □
- Pre-compute mapping (small memory cost): □
- Scalable at runtime (independent from number of objects) □

Ours	-	Yes	+	++	+

3. Contribution

3. Base idea: pre-computations at 3D grid nodes

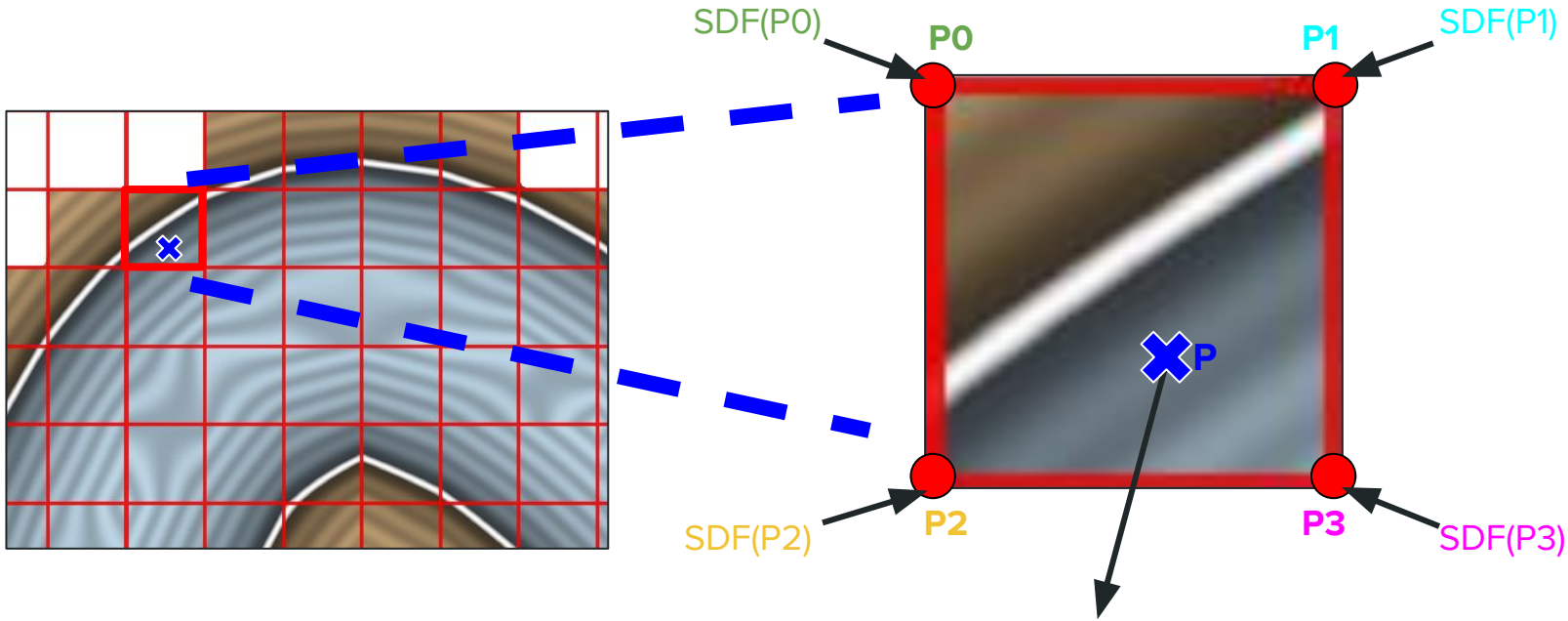




- **Pre-compute** everything expensive & **store** it at **nodes**:
  - **SDF** to retrieve the shape
  - **Mapping** to compute the density field
- Two steps:
  - Before runtime: **Pre-compute** and build structure
  - At runtime: **Reconstruct** data from structure

3. Base idea: using a 3D grid

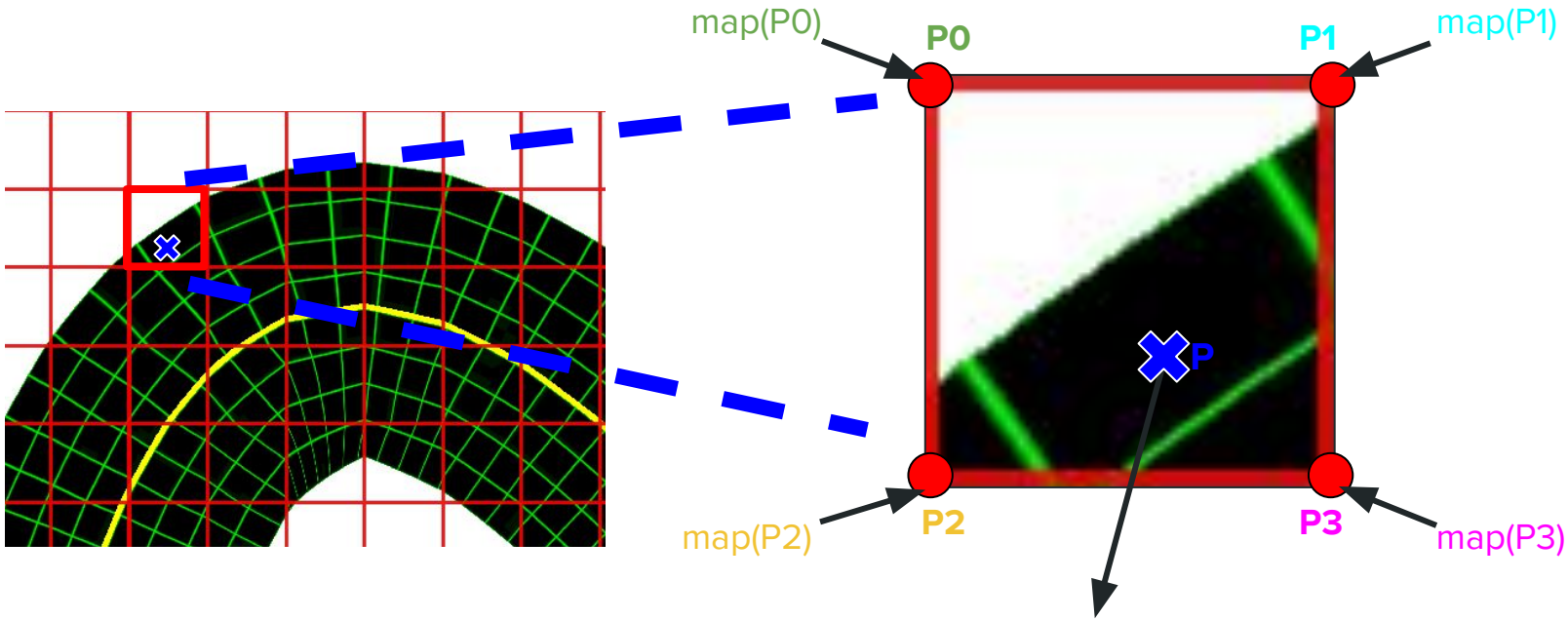
SDF reconstruction:



SDF(**P**) = Lerp (SDF(**P0**), SDF(**P1**), SDF(**P2**), SDF(**P3**), **P**)

3. Base idea: using a 3D grid

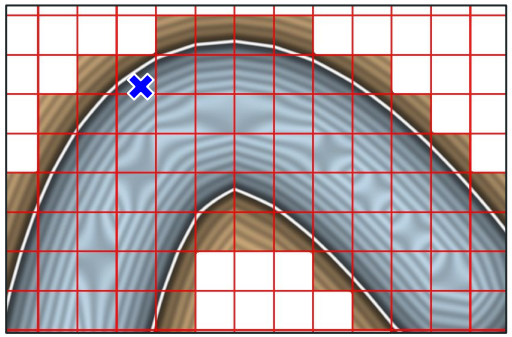
Mapping coordinates reconstruction:

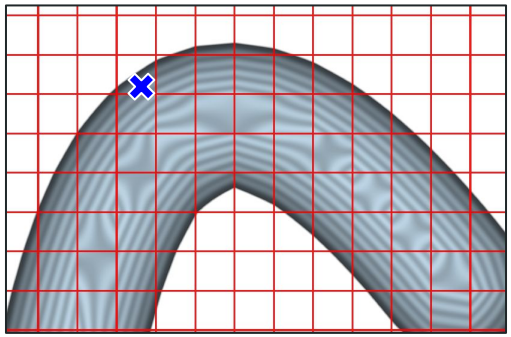


map(**P**) = Lerp (map(**P0**), map(**P1**), map(**P2**), map(**P3**), **P**)

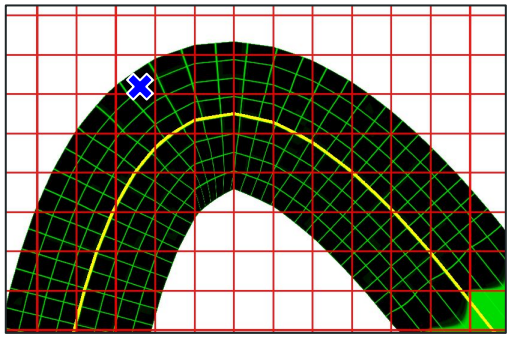
3. Base idea: using a 3D grid

SDF reconstruction: (SDF(**P**))

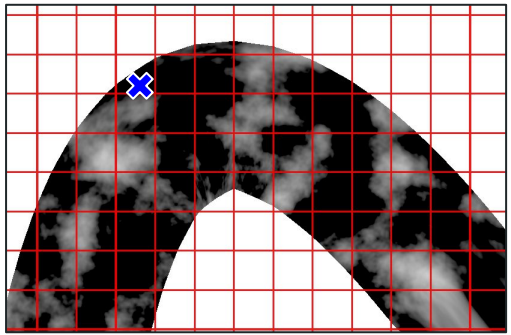


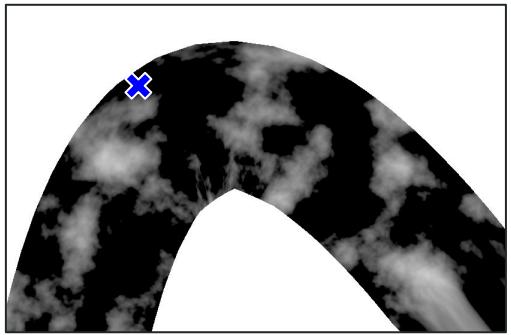


Mapping reconstruction: (map(**P**))



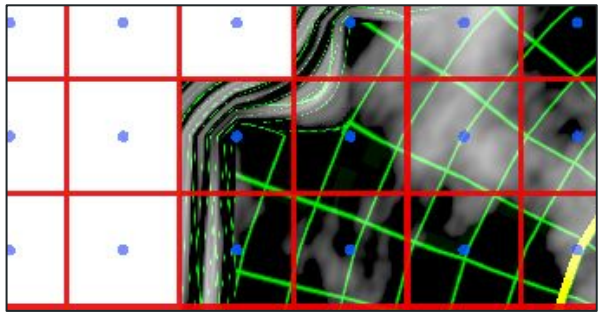
Noise evaluation: (noise(map(**P**))



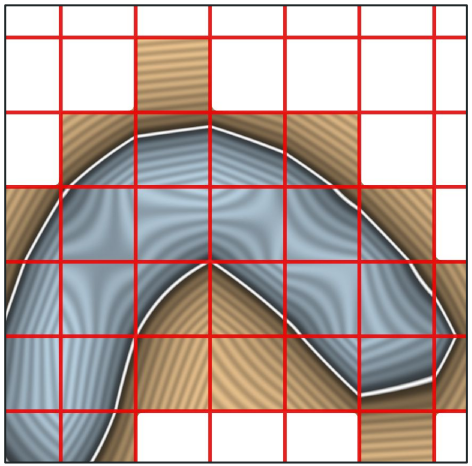


3D grid associated challenges

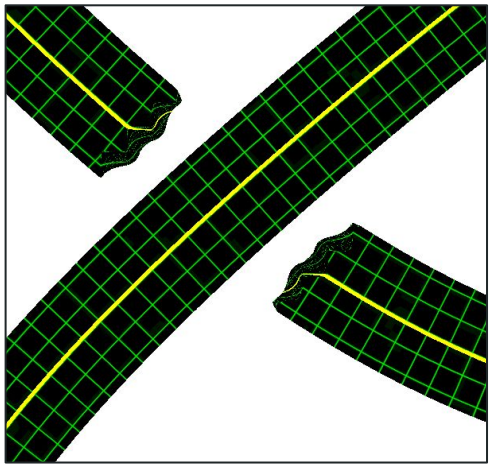
Defining valid cells



Choosing the right resolution and interpolation

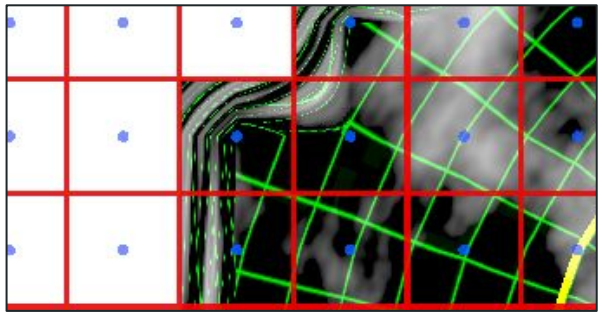


Handling superposition of objects

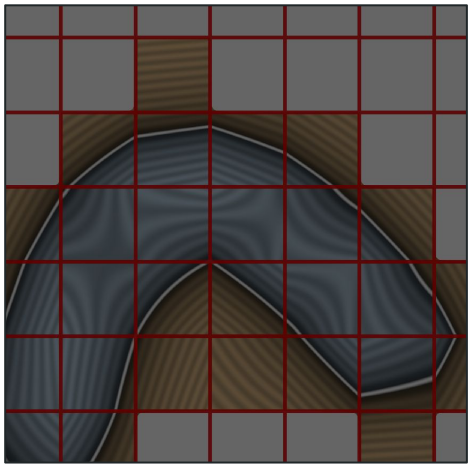


3D grid associated challenges

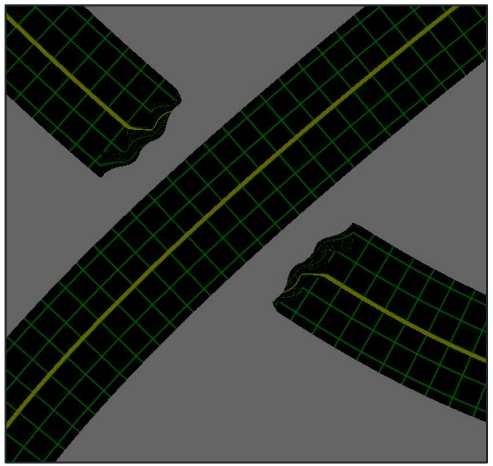
Defining valid cells



Choosing the right resolution and interpolation



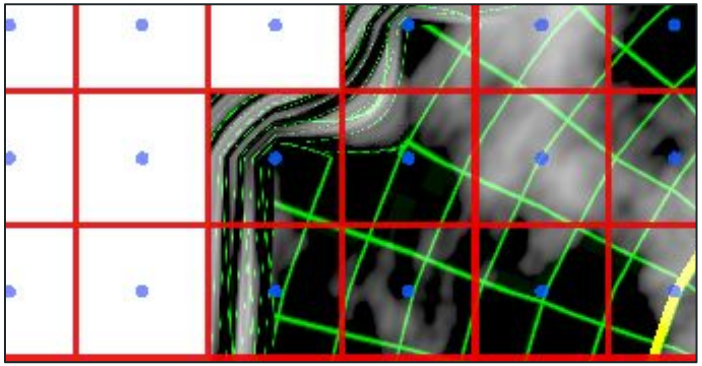
Handling superposition of objects

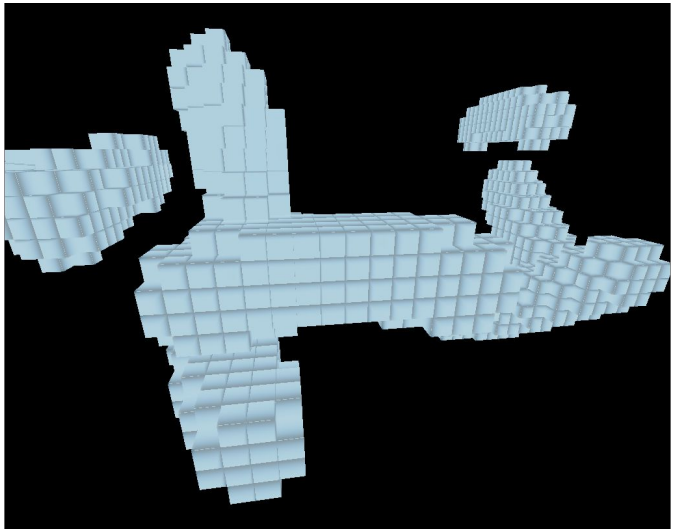


3.1. Defining valid cells

- During pre-computation:
  - **“valid” flag**
- At runtime:
  - if inside empty cell, skip
  - else, interpolation is valid
    - Interpolate SDF and mapping

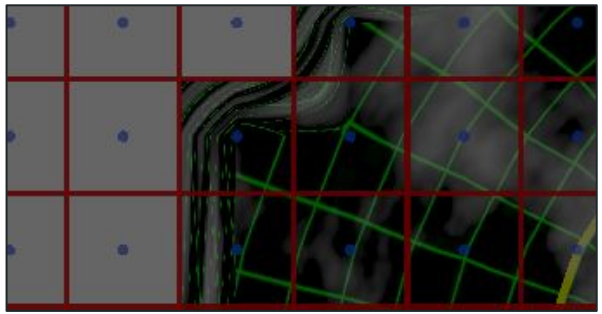
Simple way to **skip empty spaces**



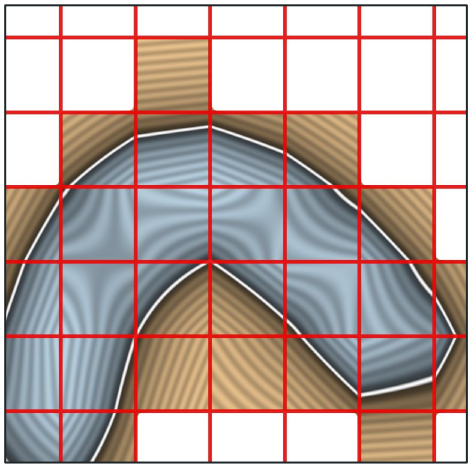


3D grid associated challenges

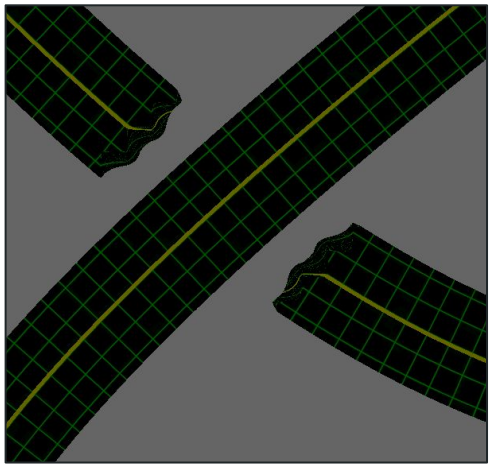
Defining valid cells



Choosing the right resolution and interpolation



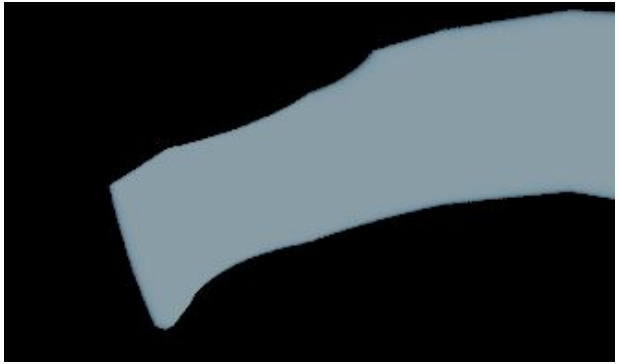
Handling superposition of objects



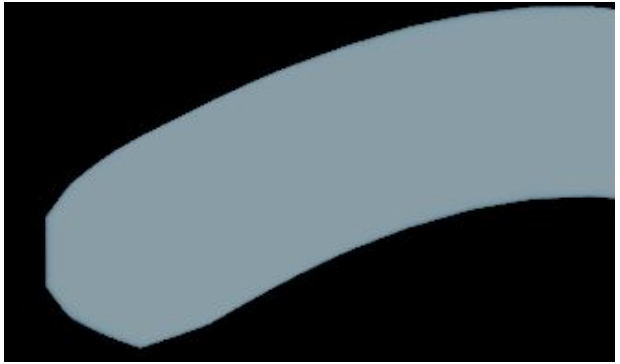
3.2. The optimal resolution: SDF

SDF reconstructions (3D):

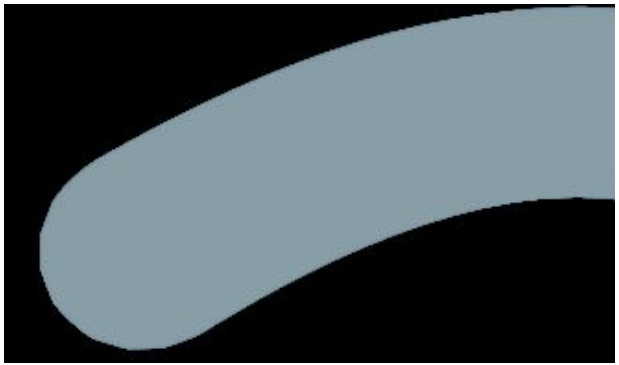
32^3 :



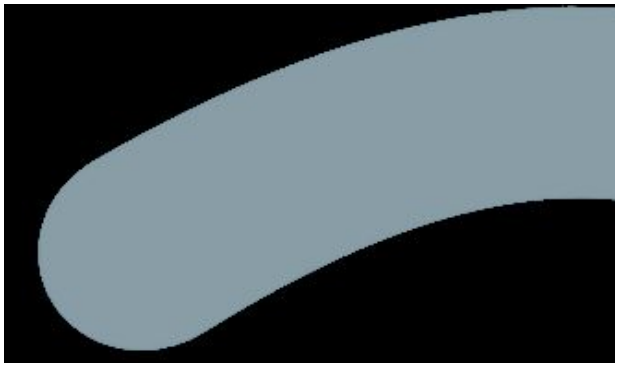
64^3 :



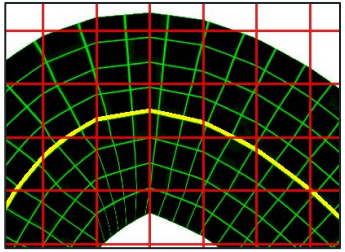
128^3 :



256^3 :

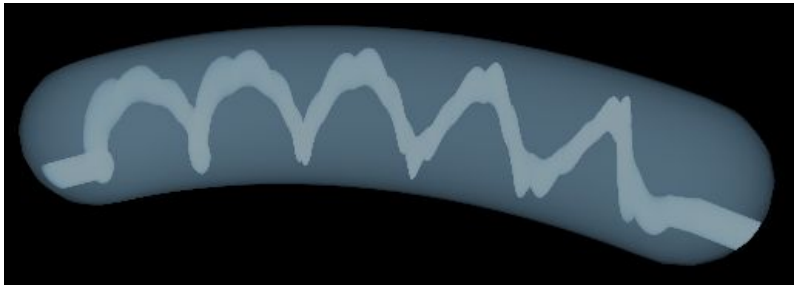


3.2. The optimal resolution: Mapping

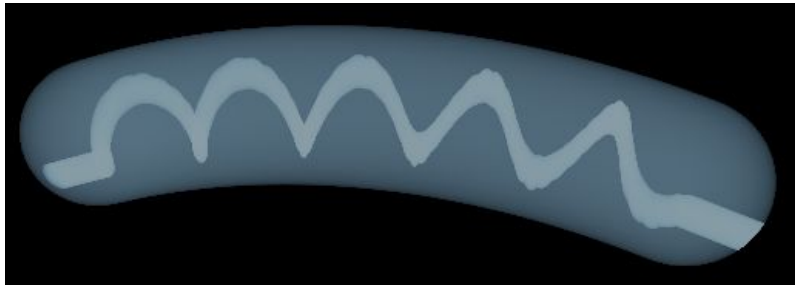


Mapping “torsion” reconstructions (3D):

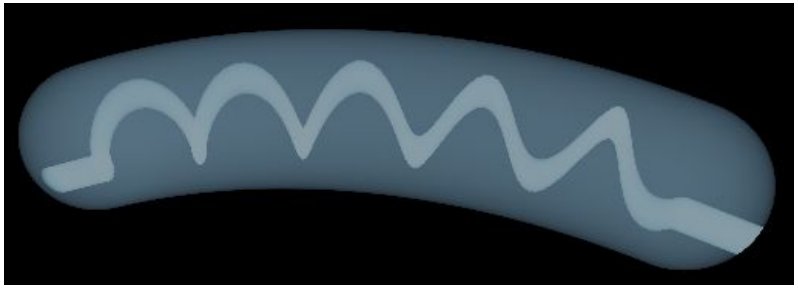
128^3 :



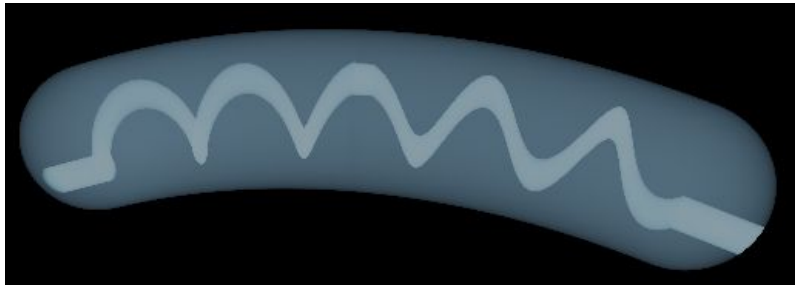
256^3 :



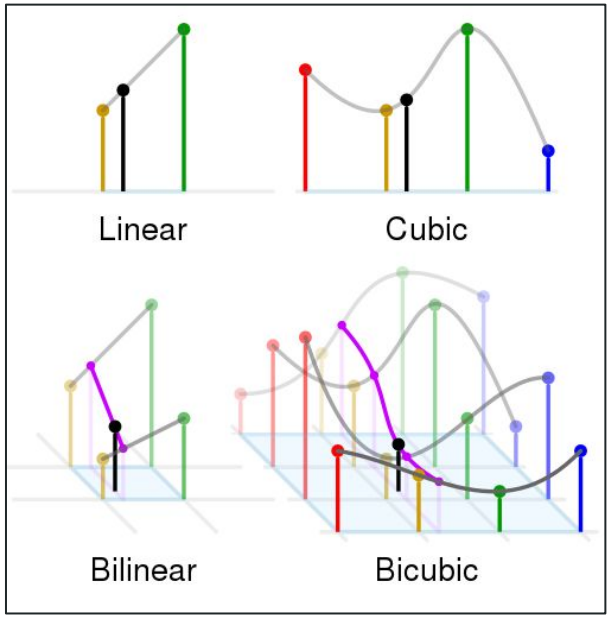
512^3 :



Exact :



3.2. Cubic interpolation



Linear: 2 samples

Bilinear: 2x2 (4) samples

Trilinear: 2x2x2 (8) samples

Cubic: 4 samples

Bicubic: 4x4 (16) samples

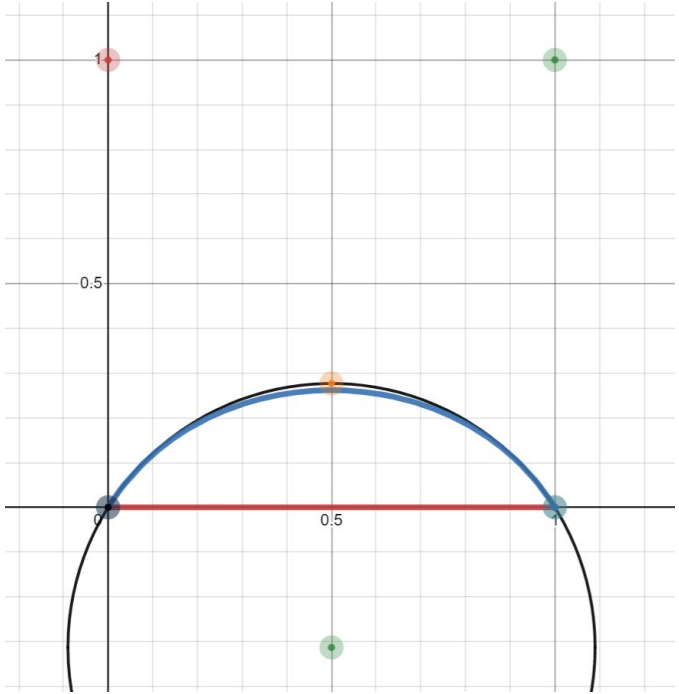
Tricubic: 4x4x4 (64) samples

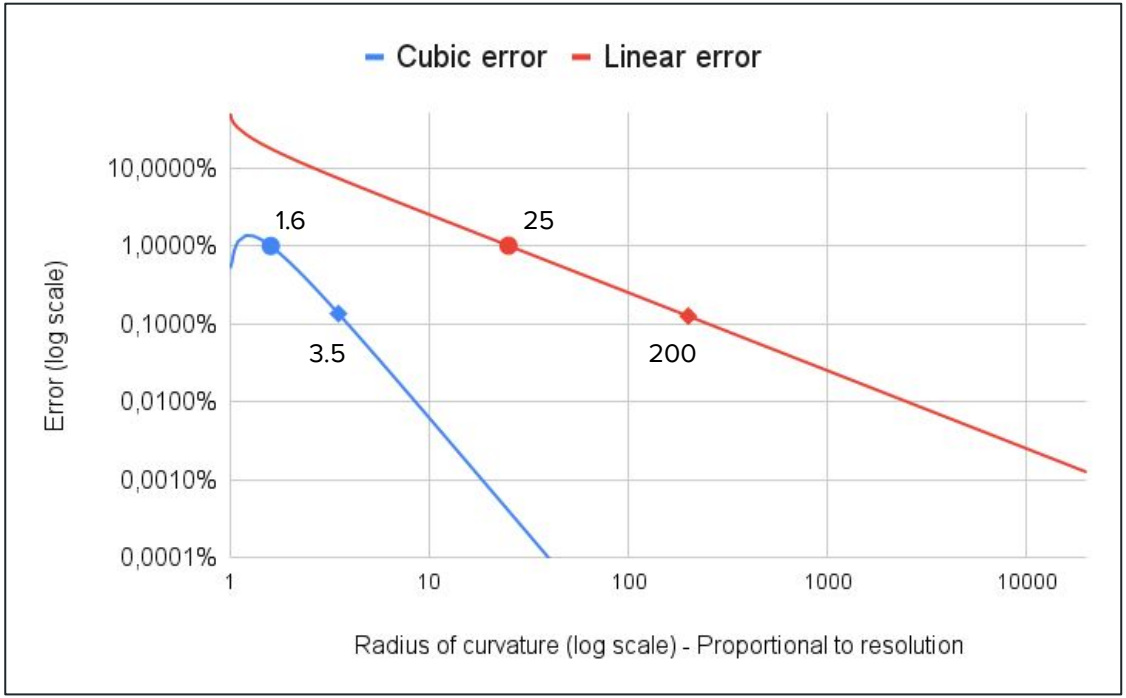
Cubic interpolation:

- is 8 times more expensive
- requires information further away

3.2. Cubic interpolation

Error depending on curvature test :



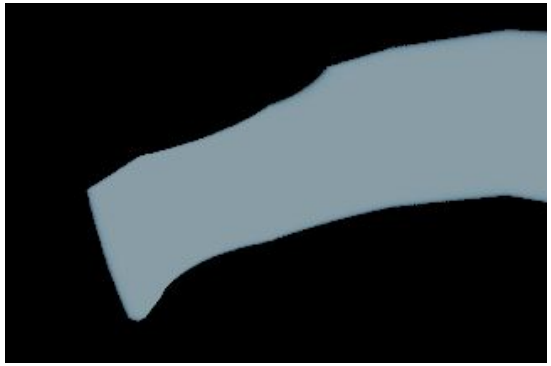


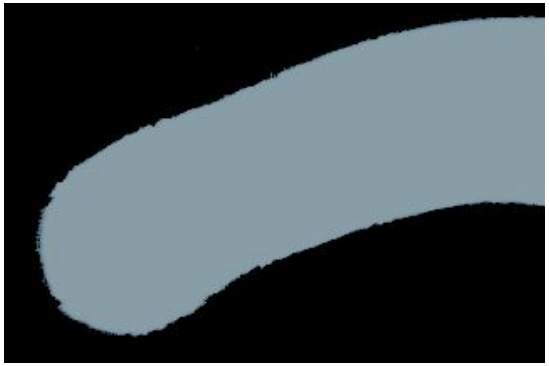
3.2. Cubic interpolation: SDF

Tri-linear :

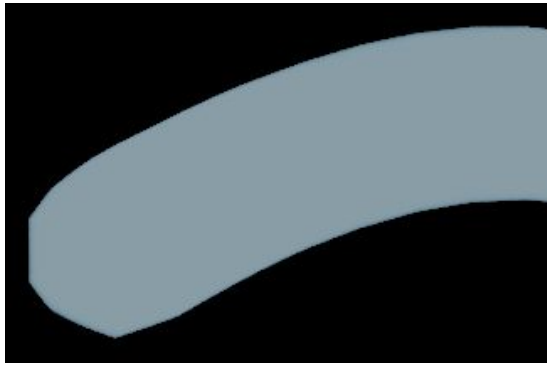
Tri-cubic :

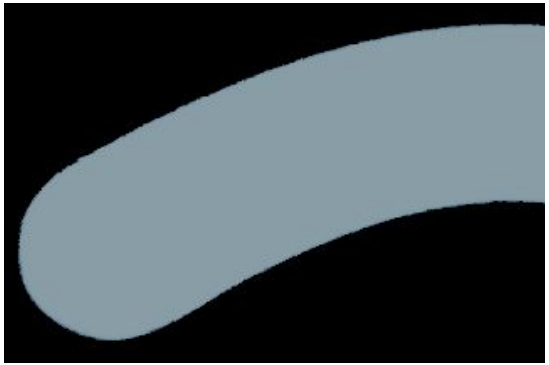
32^3 :



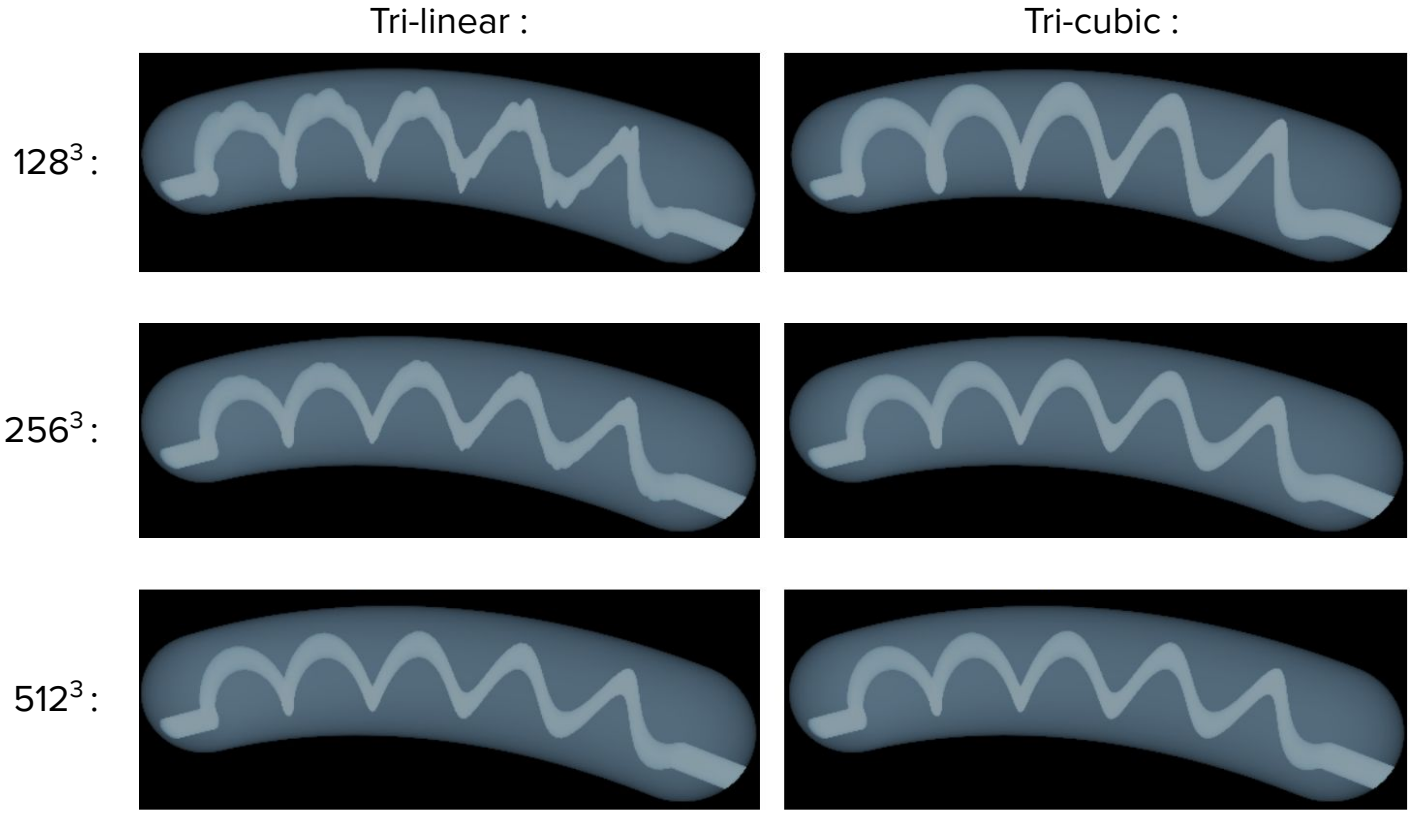


64^3 :



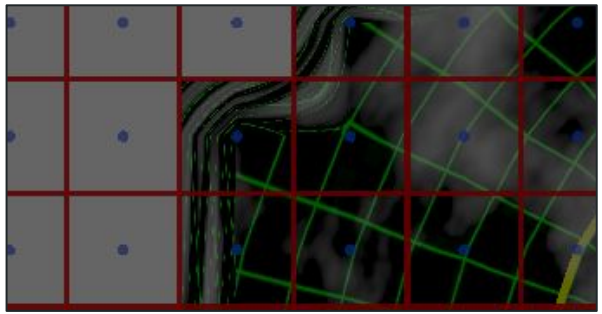


3.2. Cubic interpolation: Mapping

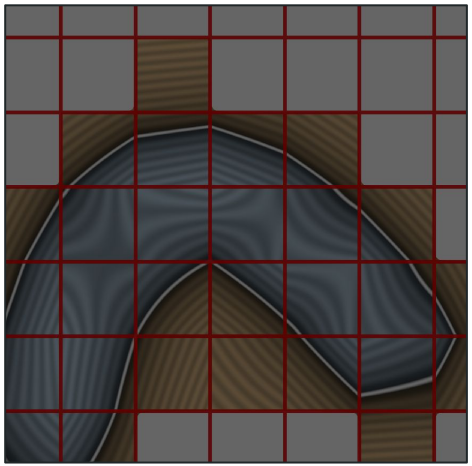


3D grid associated challenges

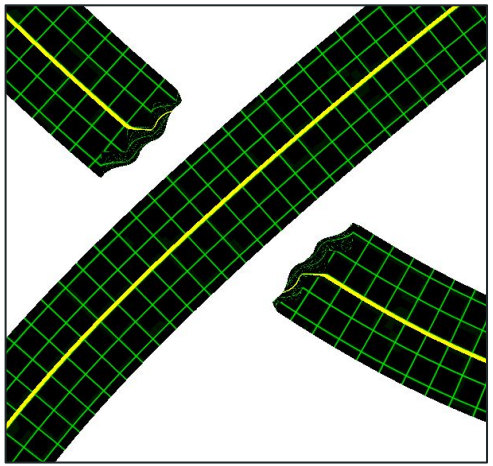
Defining valid cells



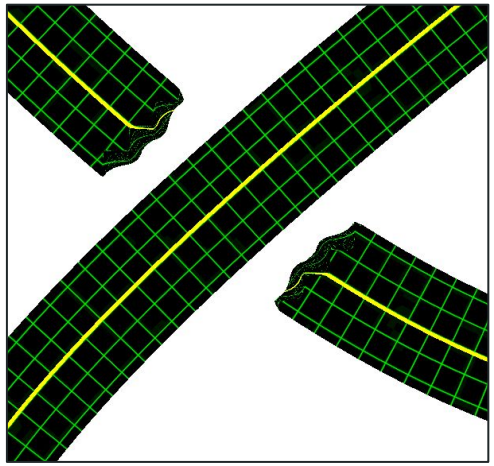
Choosing the right resolution and interpolation

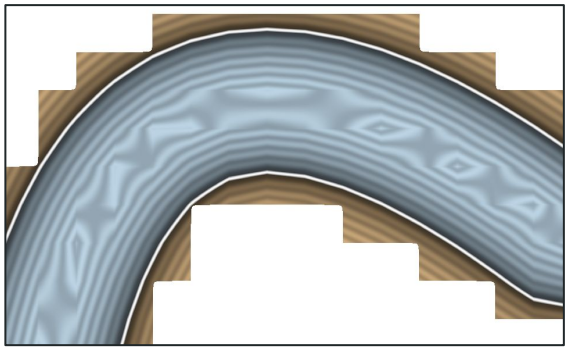


Handling superposition of objects



3.3. Overlapping objects





- Objects **footprint** are **bigger than they appear**
- They can sometime **overlap** (even if not visually)
- Grid nodes cannot store N values for N objects

Naive solution:

- One grid per object

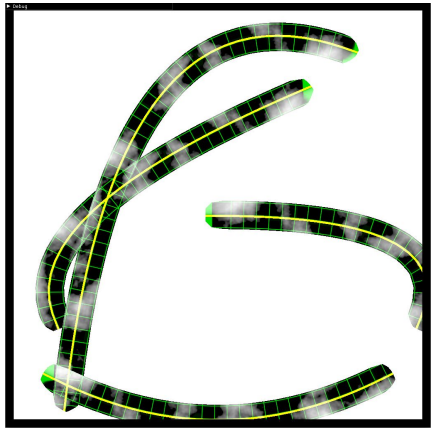
Observation : Overlap at **one place** << total number of overlaps.

Better solution:

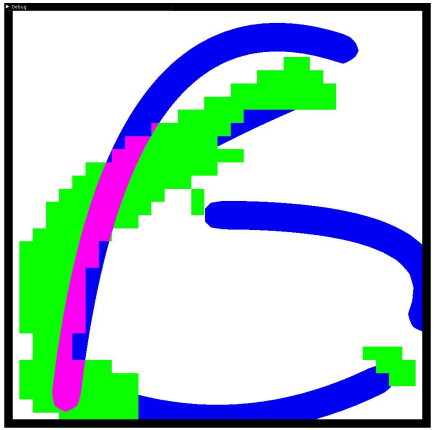
- One grid per allowed overlap at the same place

3.3. Overlapping objects

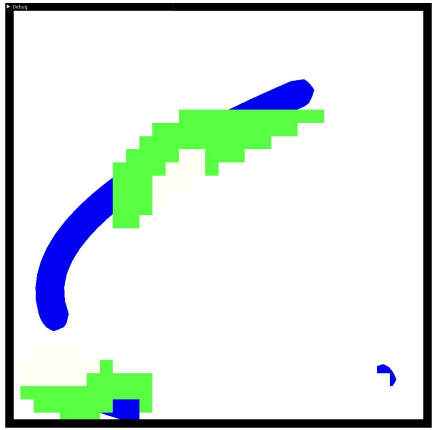
3 objects (2D - 32^3)



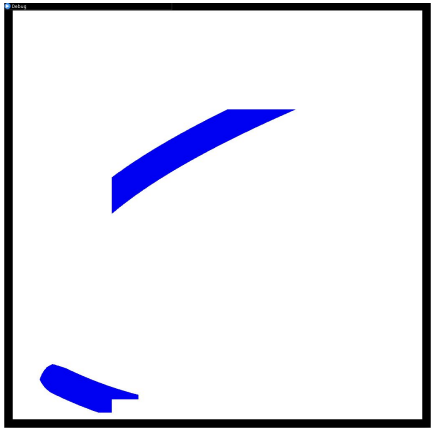
Layer 1 :



Layer 2 :



Layer 3 :



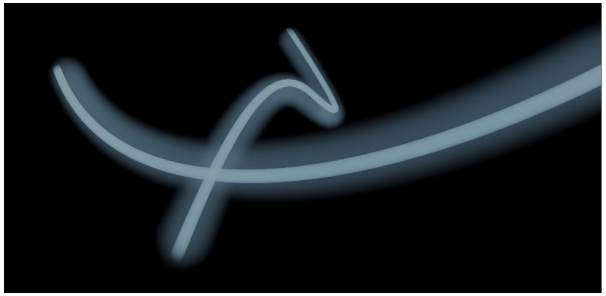
Can store most objects on the **same layer** !

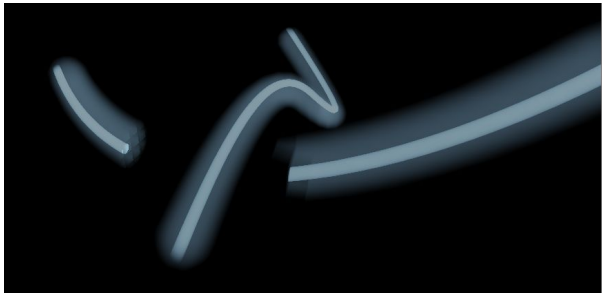
3.3. Overlapping objects

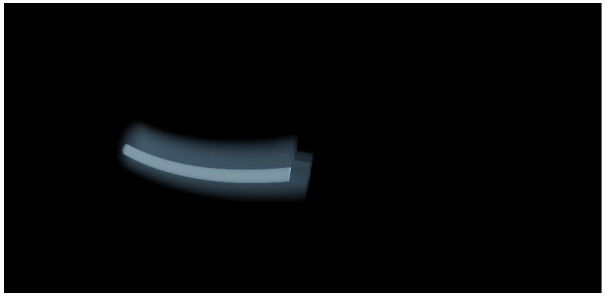
2 objects (3D - 64^3)

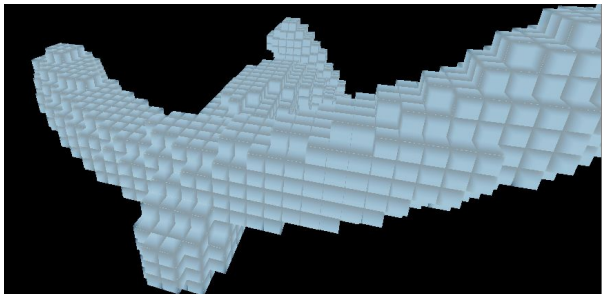
Layer 1 :

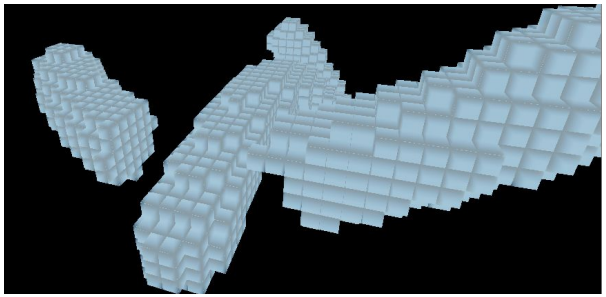
Layer 2 :

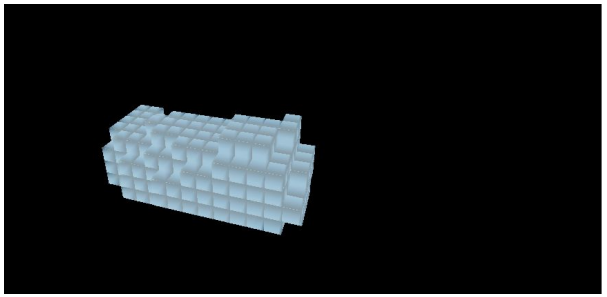








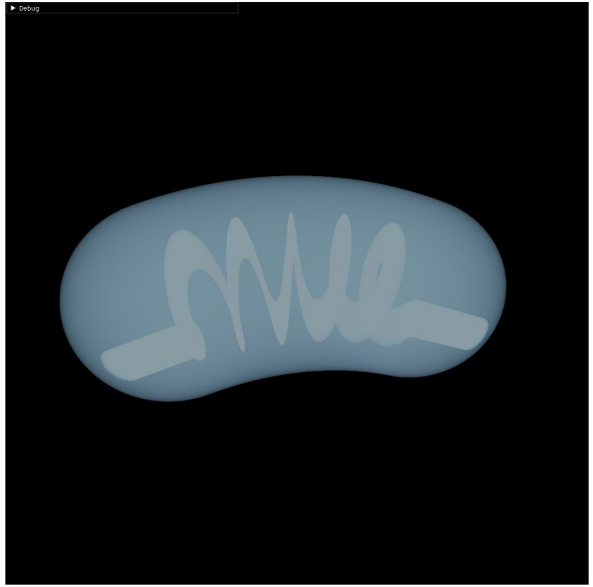




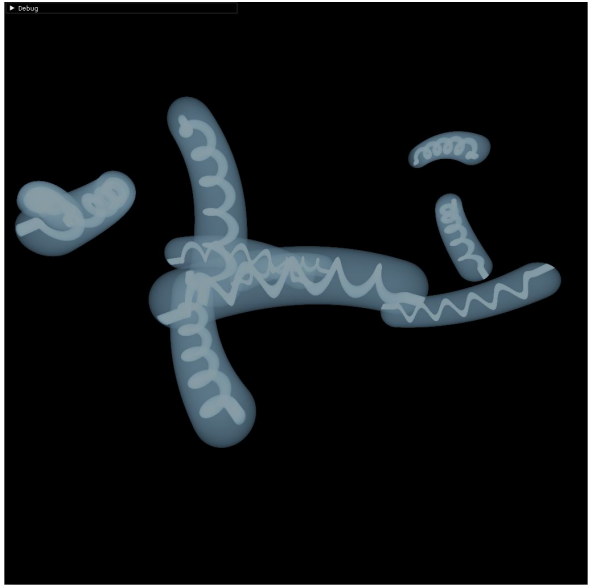
Performances & results

Performances: Test scenes

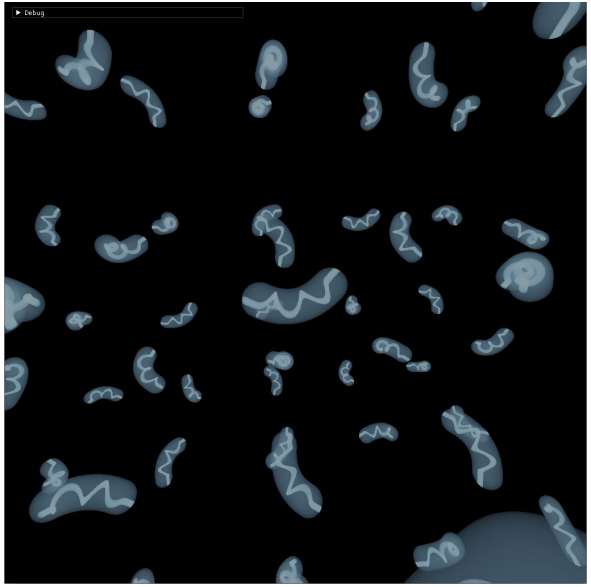
Scene N°1 (1 object) :



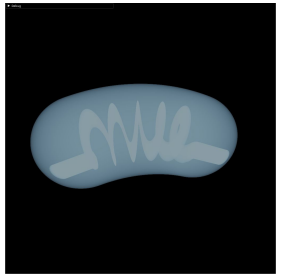
Scene N°2 (10 object) :

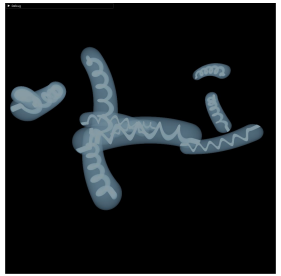


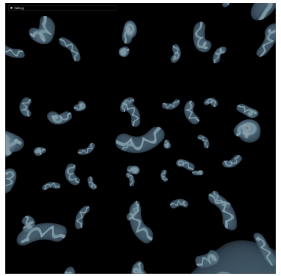
Scene N°3 (125 object) :



Performances







Performance Metrics - GTX 1080Ti (11Go Vram) - 1024x1024				
Scene	Grid resolution	Linear (ms)	Cubic (ms)	[MM22] (ms)
N°1 (1 object)	32	11.5	64.8	17.5
	64	10.4	49.7	
	128	11.4	41	
	256	18	41	
	512	24	46.6	
N°2 (10 objects)	32	9.7	31	105.1
	64	8.6	21.3	
	128	9.3	22.5	
	256	13.6	23	
	512	22	29	
N°3 (125 objects)	32	/	/	805
	64	/	/	
	128	9	20.6	
	256	13.5	21.6	
	512	/	/	

Performances: pre-computation

Scene	Grid resolution	Pre-computation (ms)	[MM22] (ms)
N°1 (1 object)	32	0.2	17.5
	64	0.9	
	128	5.2	
	256	22.2	
	512	731	
N°2 (10 objects)	32	1	105.1
	64	5.6	
	128	42	
	256	326.4	
	512	2678	
N°3 (125 objects)	32	8.3	805
	64	60.3	
	128	501	
	256	3996.5	
	512	/	

We used **implicit volumes** (spline tubes)

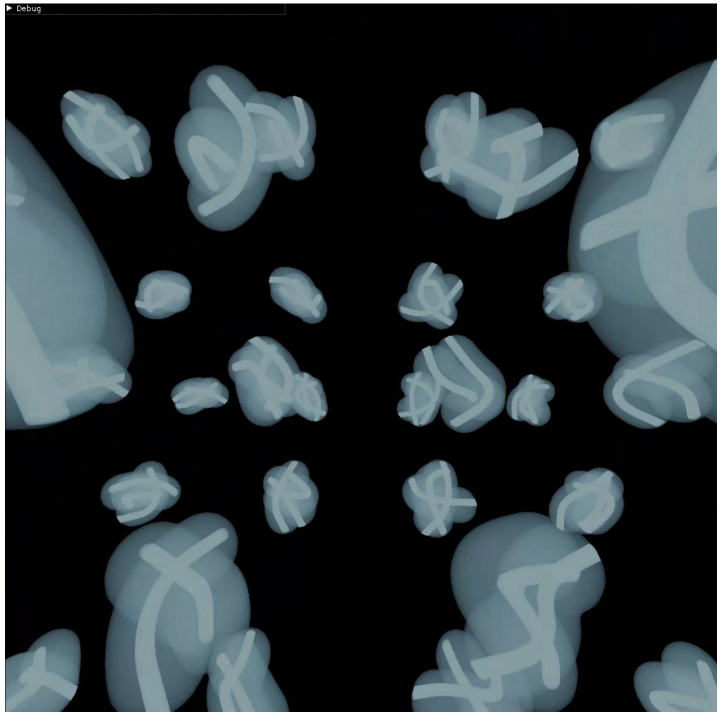
Performances ➜ At reasonable quality, could refresh grid often

Compatible with infinite scene ?

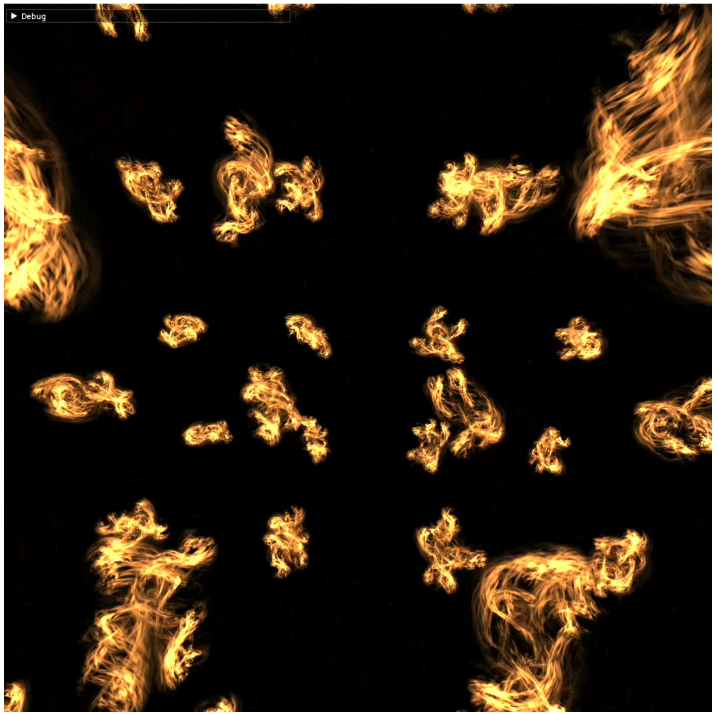
- On the fly generation

Results

150 spline tubes ($128^3 \approx 8$ MB):

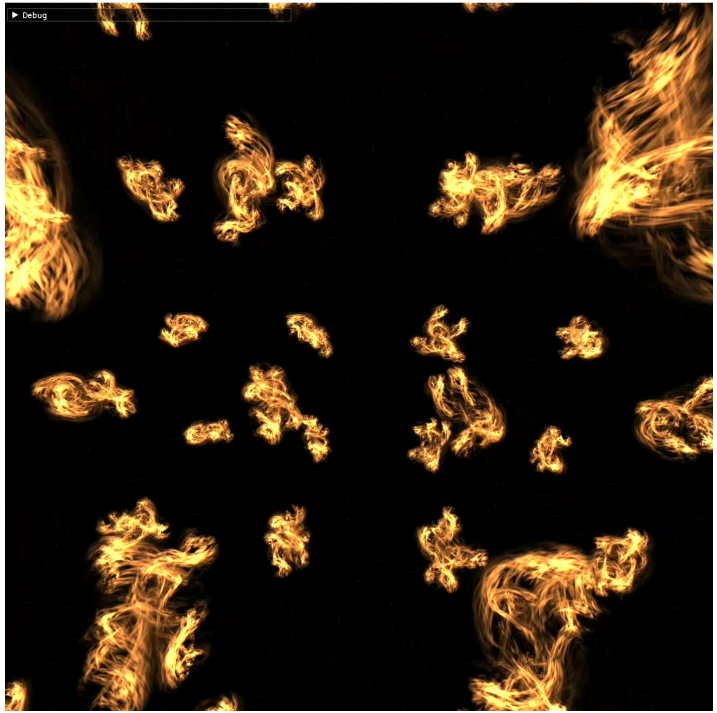


“Solar flare” material applied:

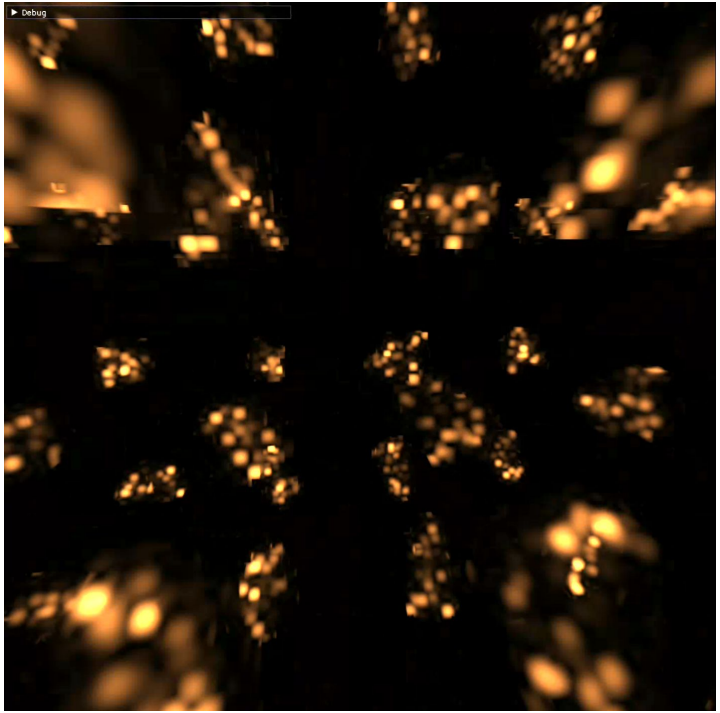


Results

“Solar flare” material: Ours (50 fps)/ [MM22] (1 fps)



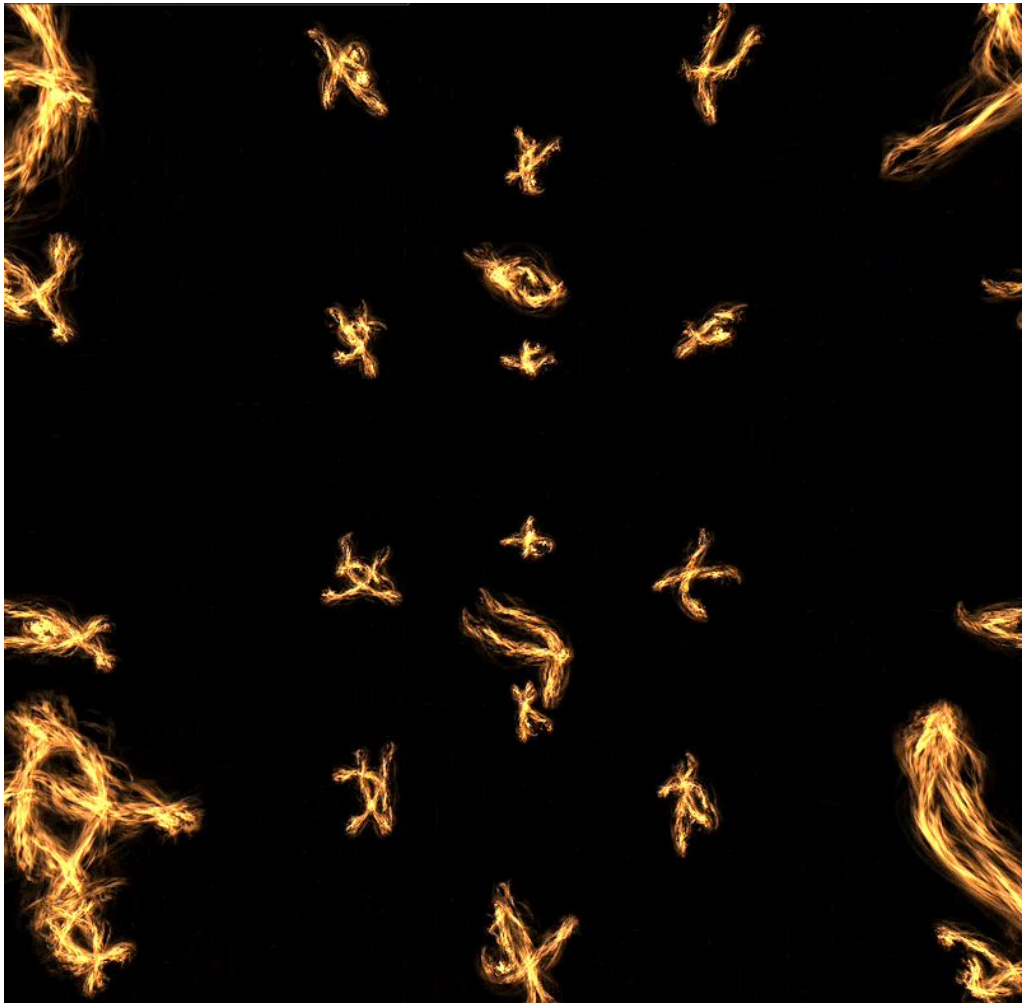
“Solar flare” material: Density directly (15 fps)



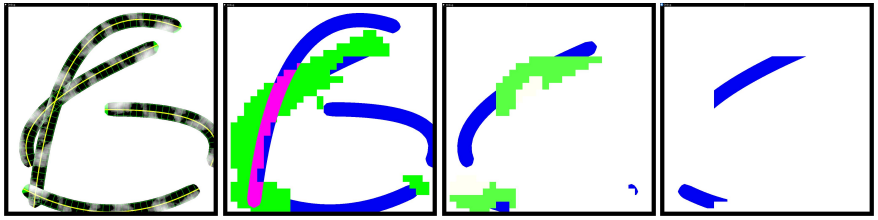
Results

Animated “solar flare” material:

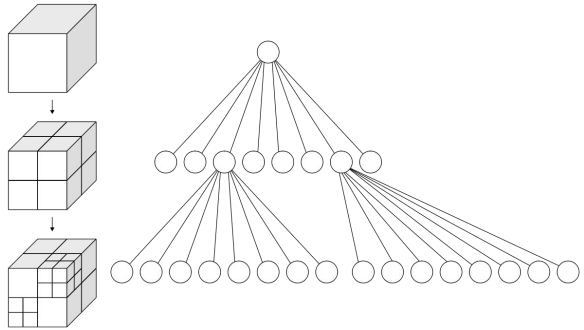
- 150 objects (up to 3 overlaps)
- 128^3 grid (≈ 8 MB)
- Linear interpolation
- 50 fps (1024x1024 image)



Future work



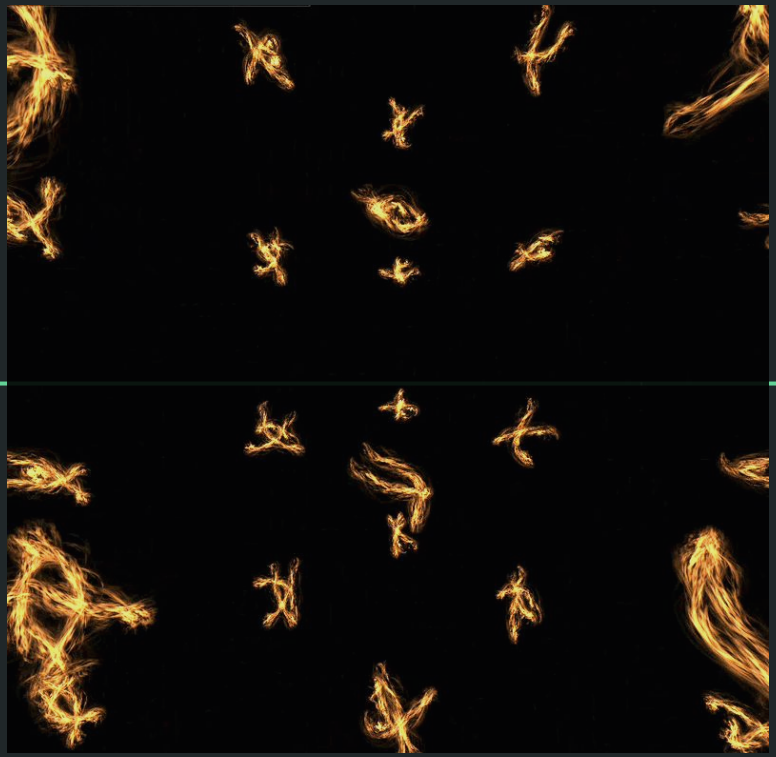
- Memory optimizations
  - sparse texture
  - octree
- Explore other construction methods:
  - “Painting” the grid values
  - Particle system



Thank you for your attention.

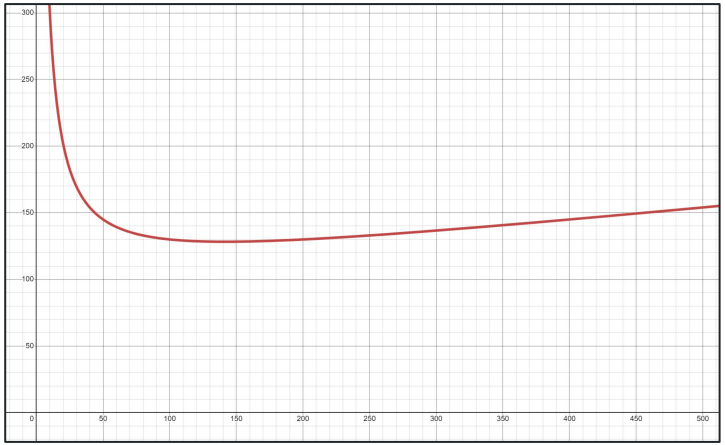
Animated “solar flare” material:

- 150 objects
- 128^3 grid (8 MB)
- 50 fps (1024x1024 image)



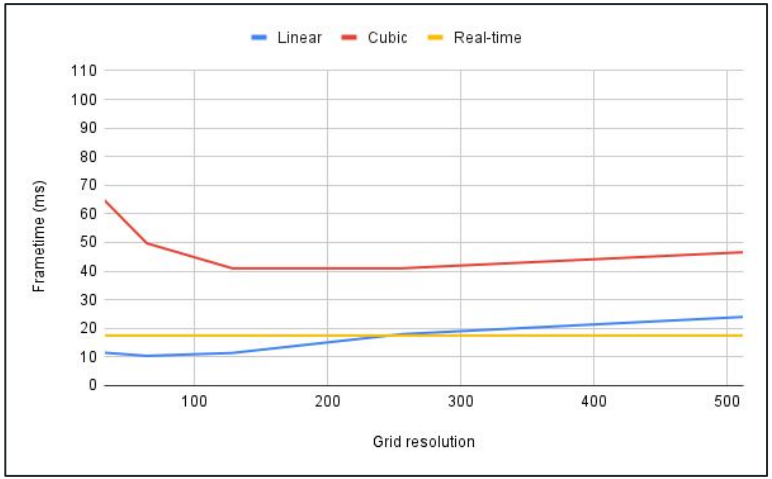
Vol ray casting cost

Theoretical performances :

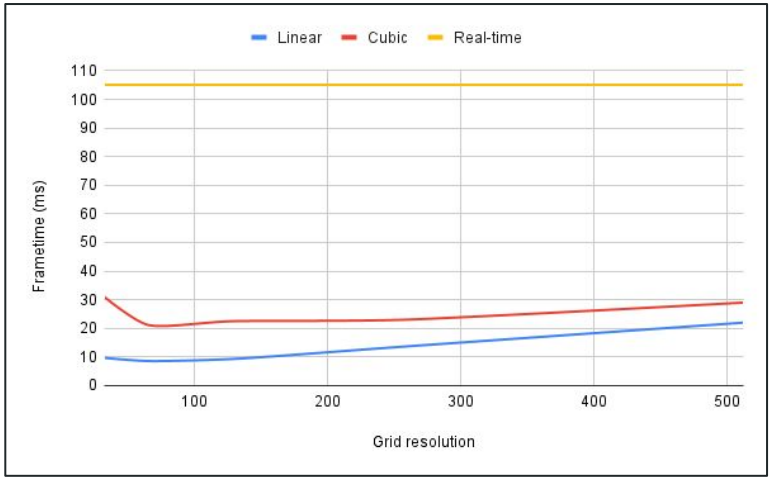


Measured performances :

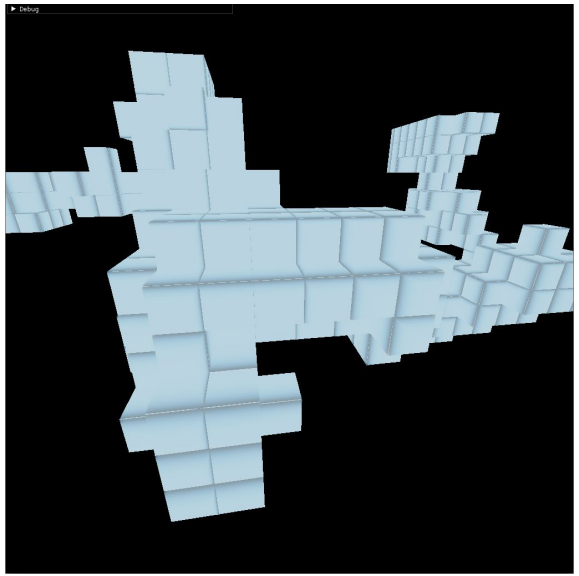
Scene 1 :

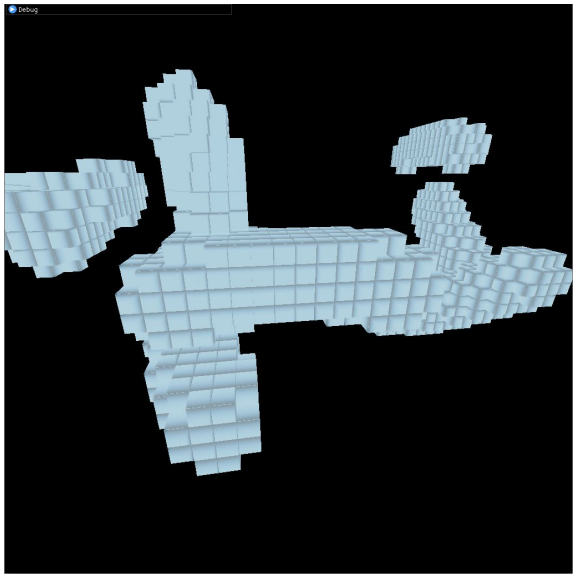


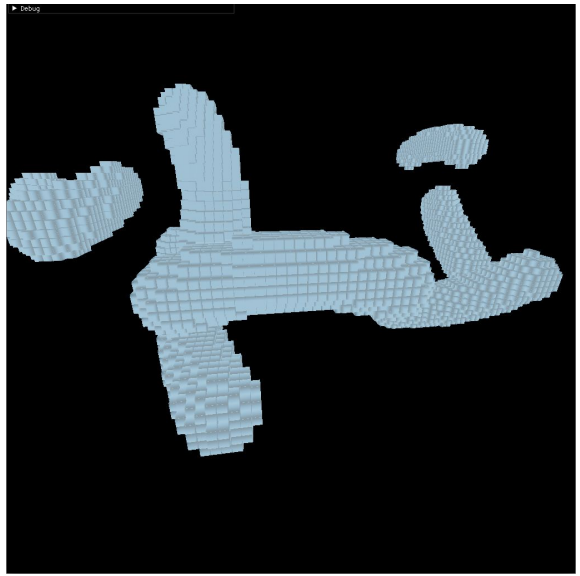
Scene 2 :



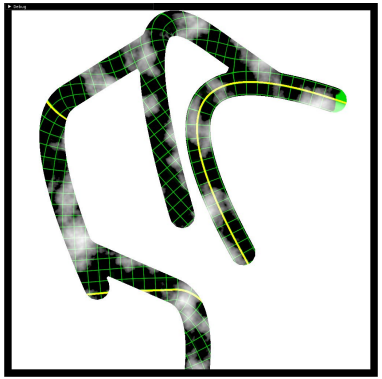
Bounding volume depending on resolution

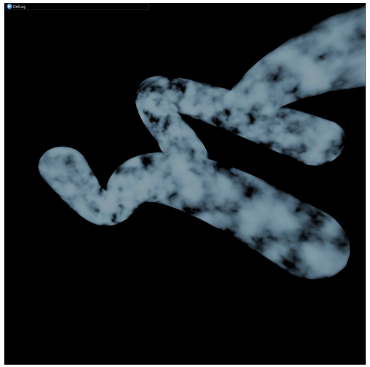


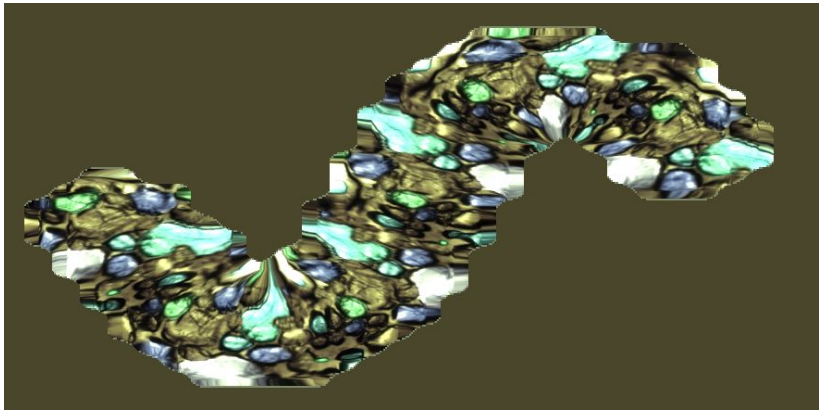




Representable objects







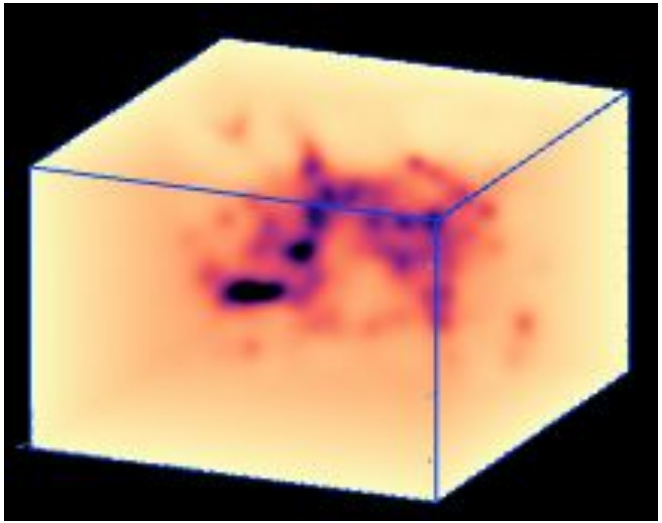
- Implicit volumes
  - spline tubes, spline trees
- “Painting”
- Mesh based
- Physic based simulation

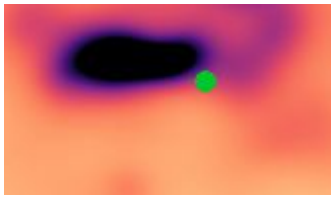
Extreme lack of reference data

Orion nebula (M42) (≂25ly):

Orion Molecular Cloud Complex (≂1000-1500ly):

[Dharmawardena, et al., 2022]





2D splatting