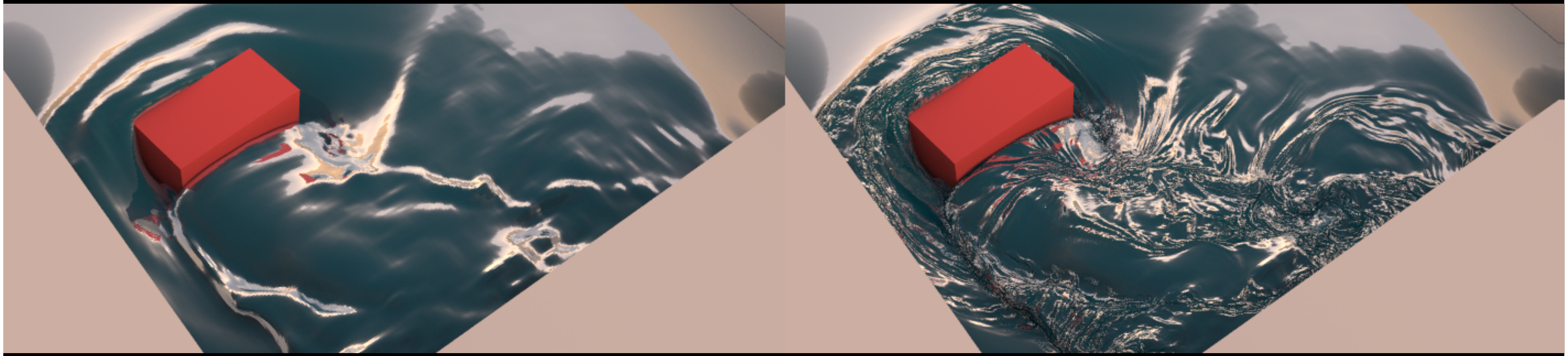
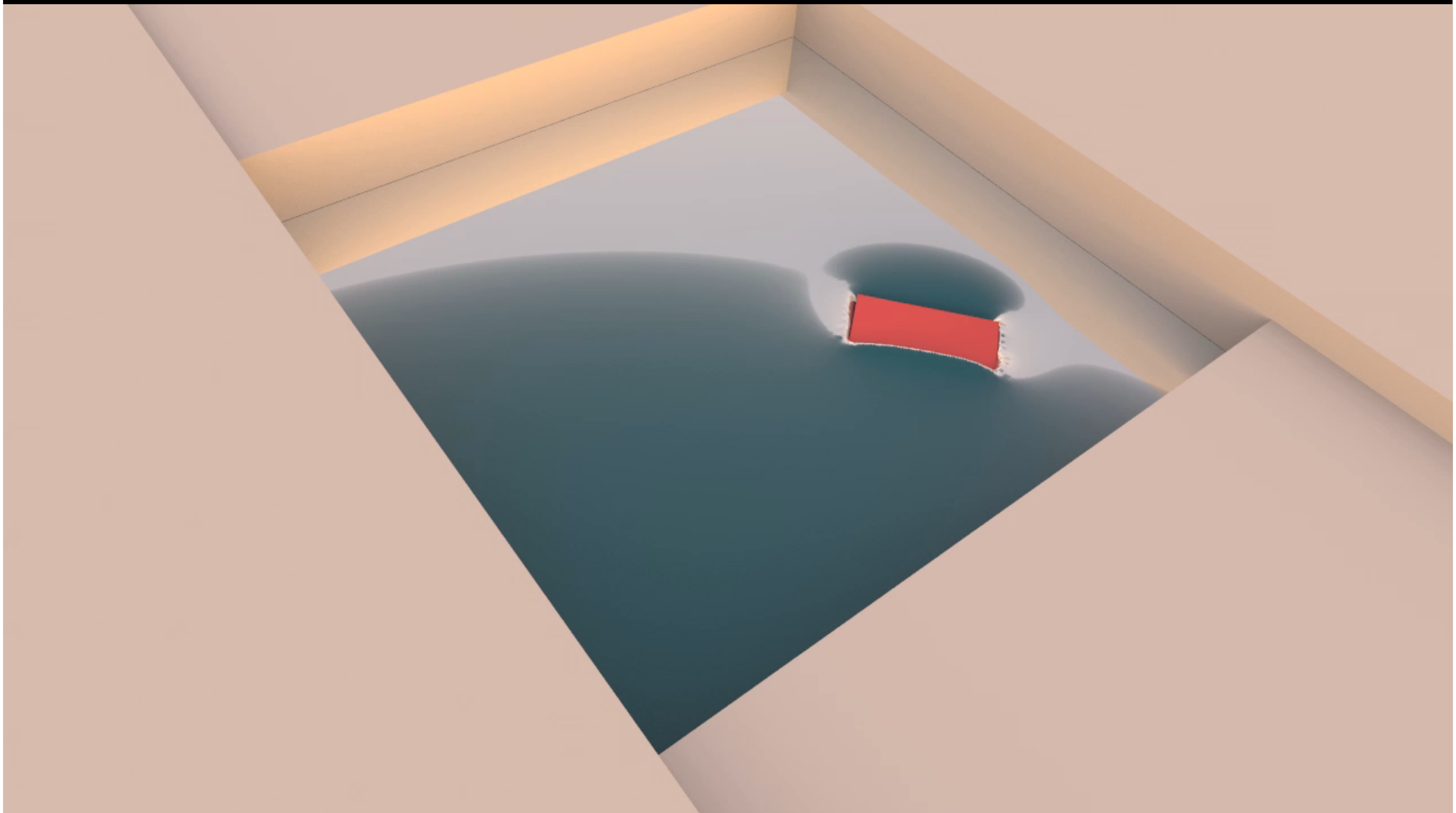


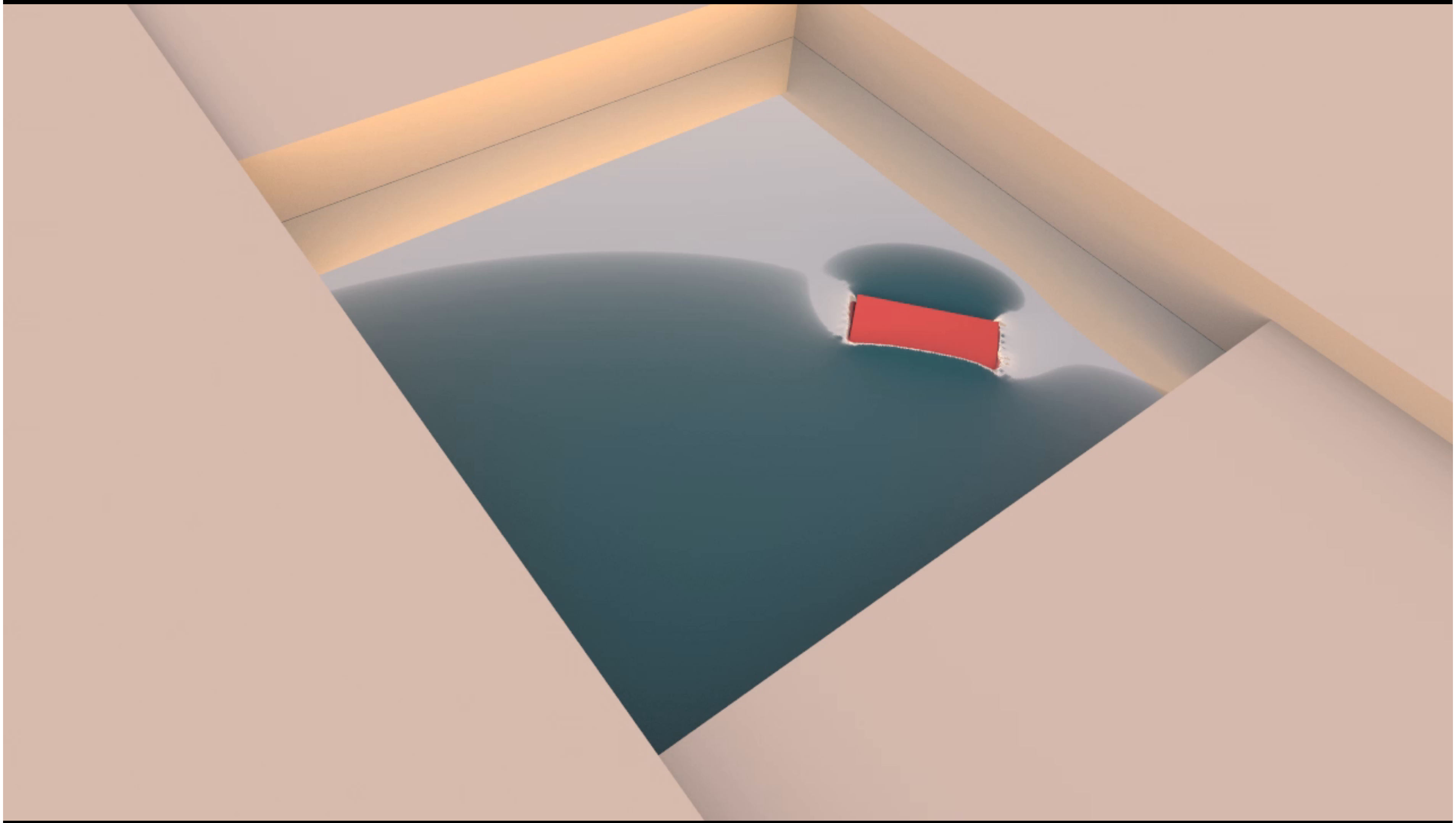
Closest Point Turbulence for Liquid Surfaces

Theodore Kim, Nils Thuerey, Jerry Tessendorf

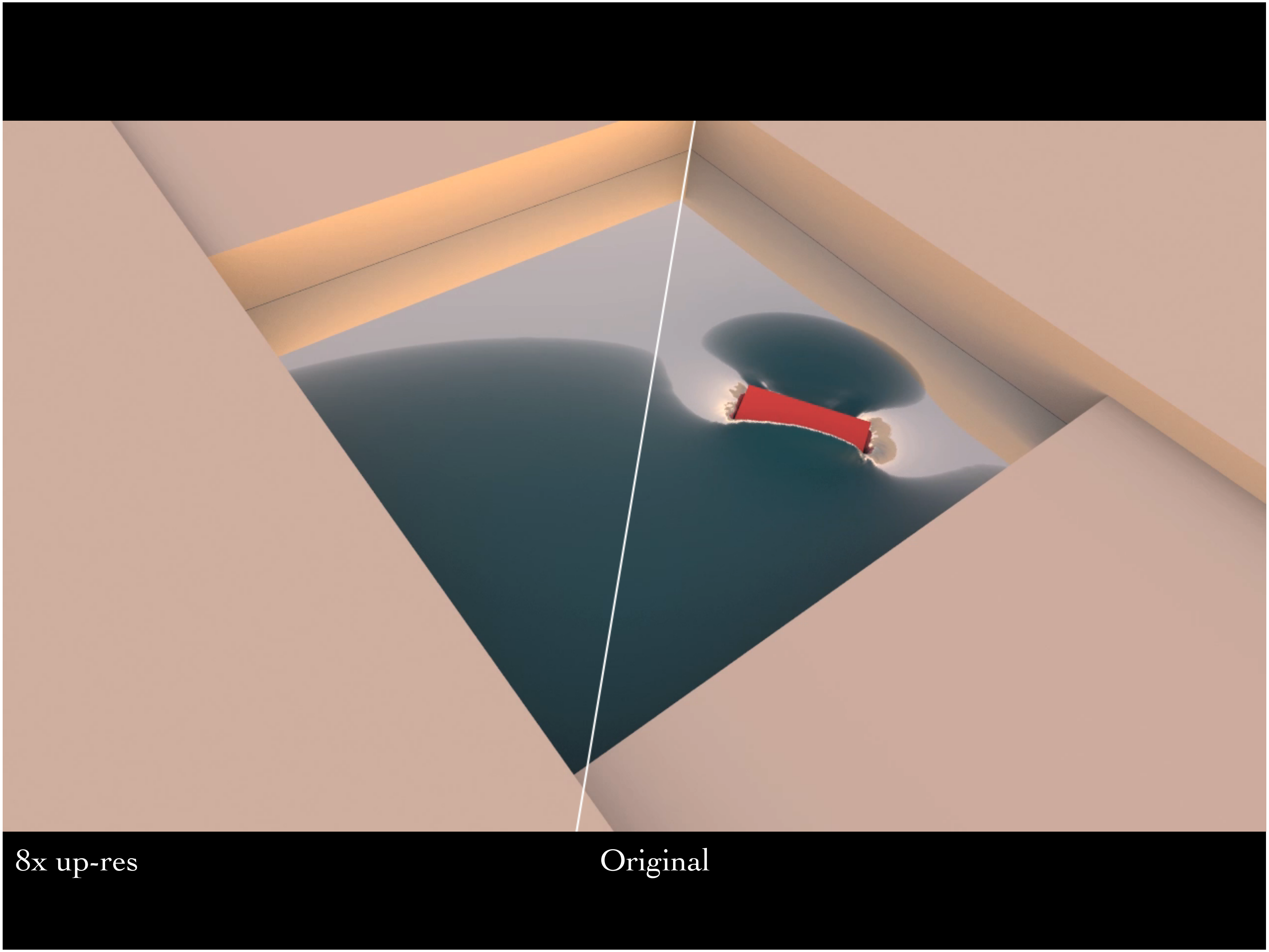
ACM Transactions on Graphics, 2013



100^3 Houdini simulation



8x up-res, 100^3 Houdini simulation


8x up-res
Original

Outline

- Previous Works
- The Closest Point Method
- 3D iWave
- Additional Extensions
- Results

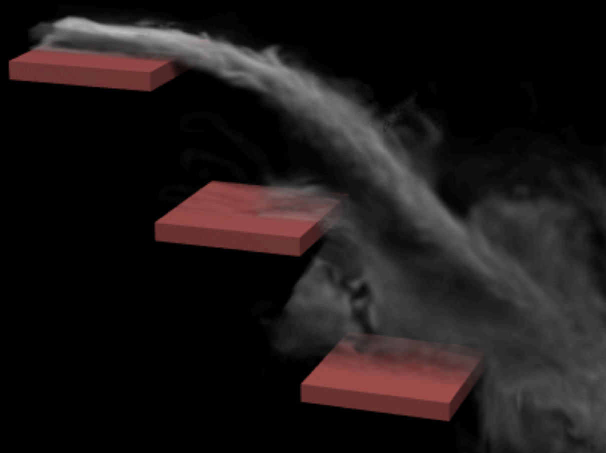
Outline

- Previous Works
- The Closest Point Method
- 3D iWave
- Additional Extensions
- Results

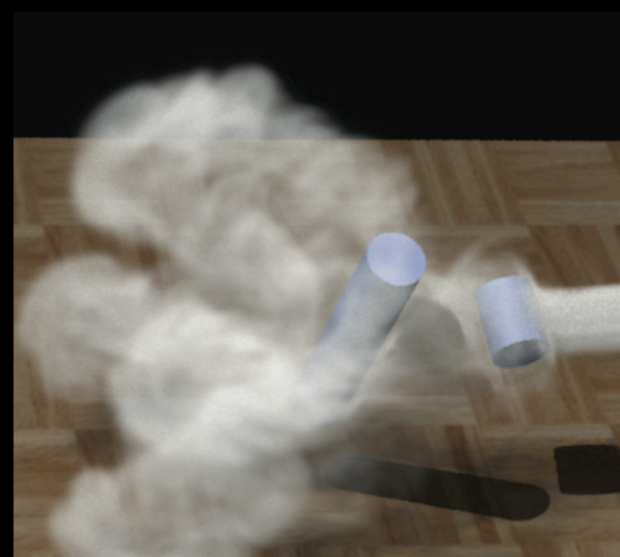
Smoke Up-Res

[Kim et al. 2008]

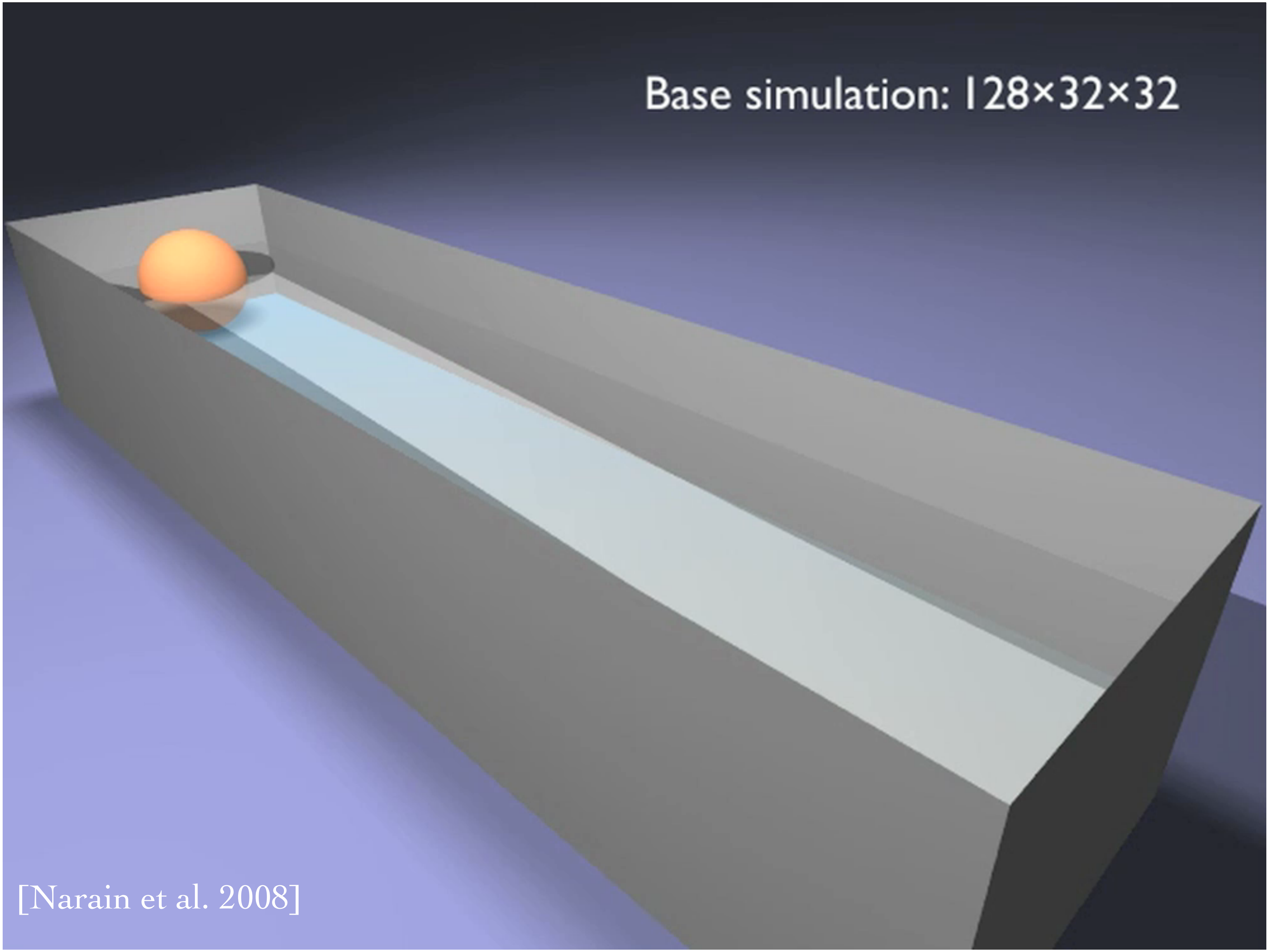
[Nielsen et al. 2009]



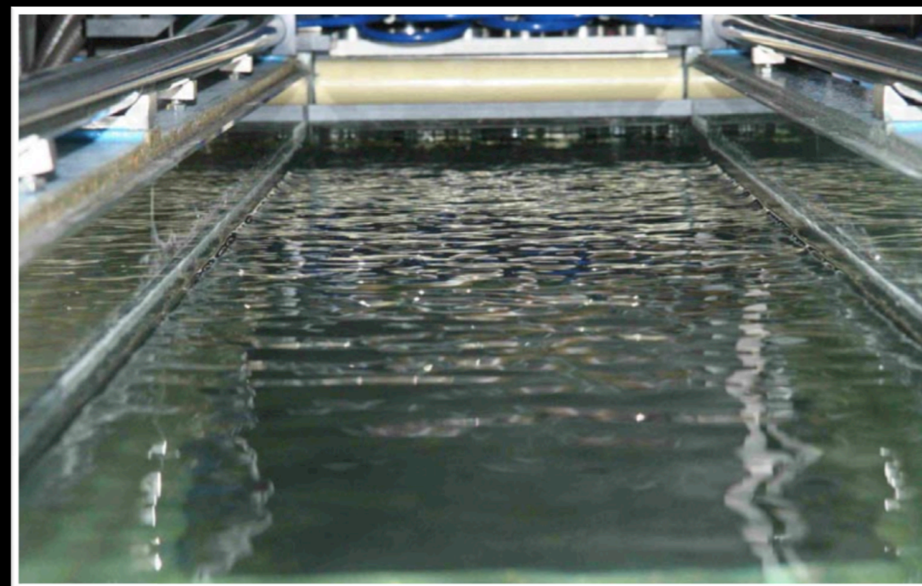
[Huang et al. 2011]



[Yuan et al. 2011]


Base simulation: 128×32×32
[Narain et al. 2008]

[Falcon 2010]



[Savelsberg and van de Water 2008]

PRL **100,** 034501 (2008) PHYSICAL REVIEW LETTERS week ending 25 JANUARY 2008

Turbulence of a Free Surface

Ralph Savelsberg and Willem van de Water
Physics Department, Eindhoven University of Technology, Postbus 513, 5600 MB Eindhoven, The Netherlands
(Received 9 August 2007; published 23 January 2008)

Surface waves travel *much faster* than the underlying water

Free Surface Turbulence

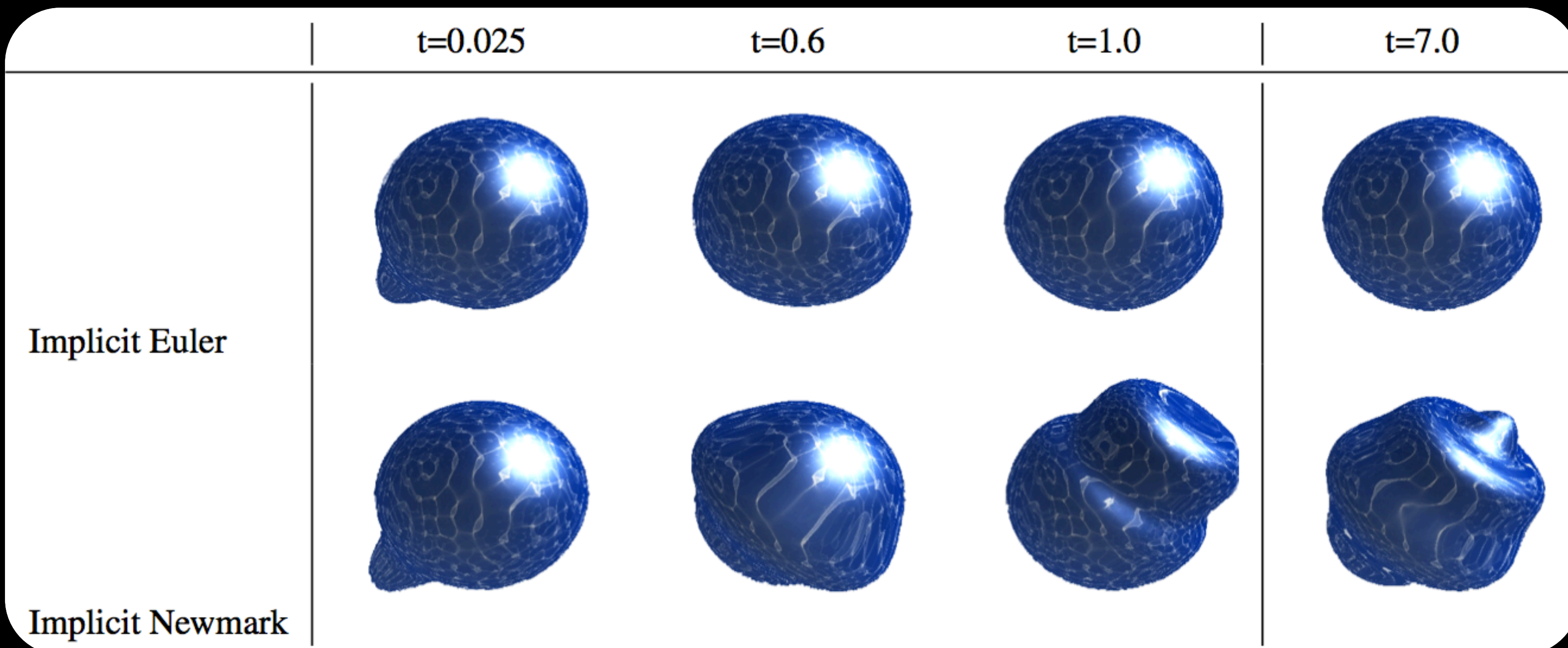
- Kolmogorov spectrum: $k^{-\frac{5}{3}}$
- Kolmogorov-*Zakharov* spectrum: $k^{-\frac{11}{4}}$
  - A.k.a. "wave" or "weak" turbulence
  - [Zakharov 1968]

Free Surface Turbulence

- The approach
  - Low resolution simulation *initiates* waves
  - Run a *wave simulation* on the liquid surface
  - All purely Eulerian

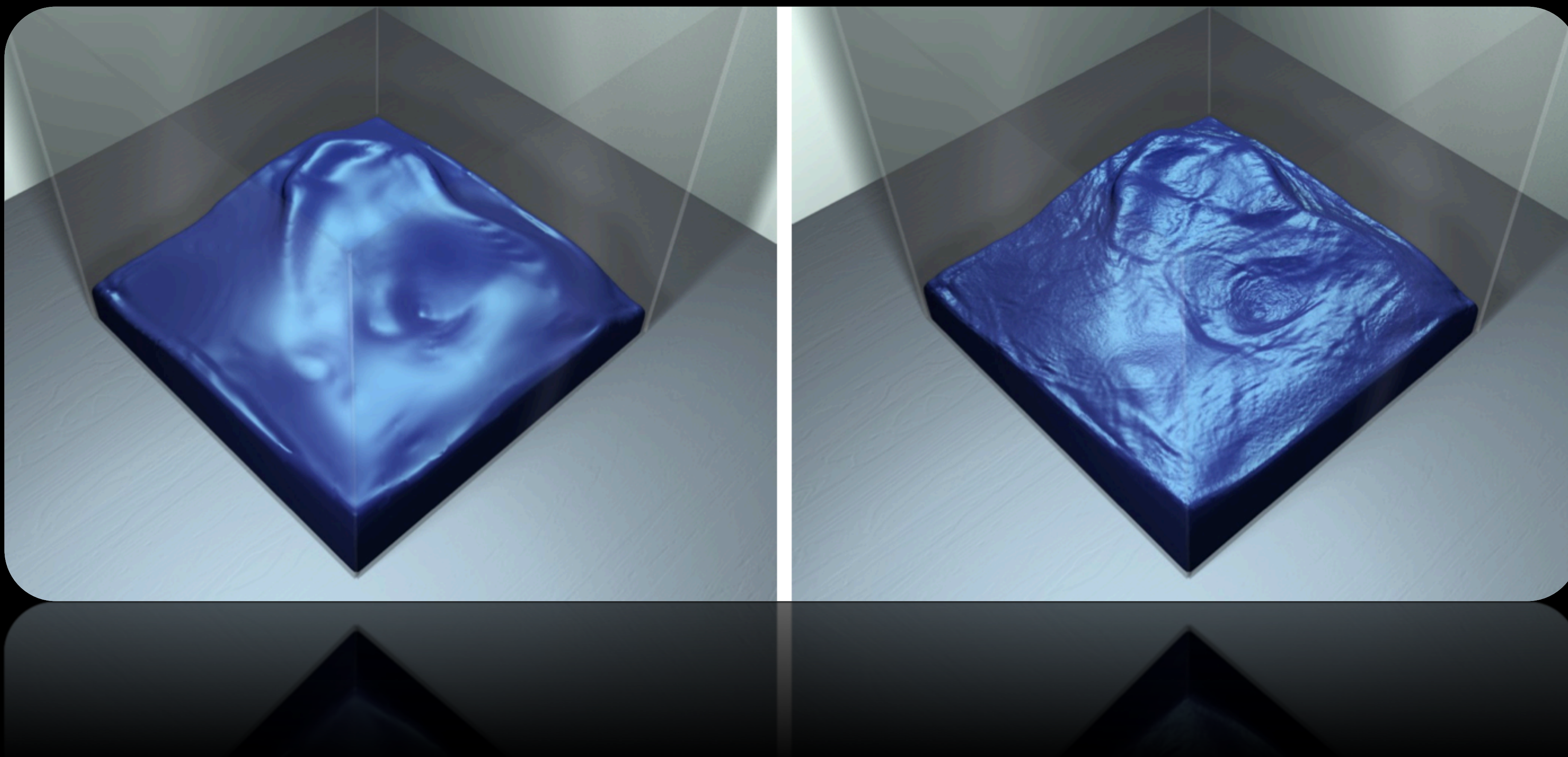
Surface Simulation

[Angst et al. 2008]



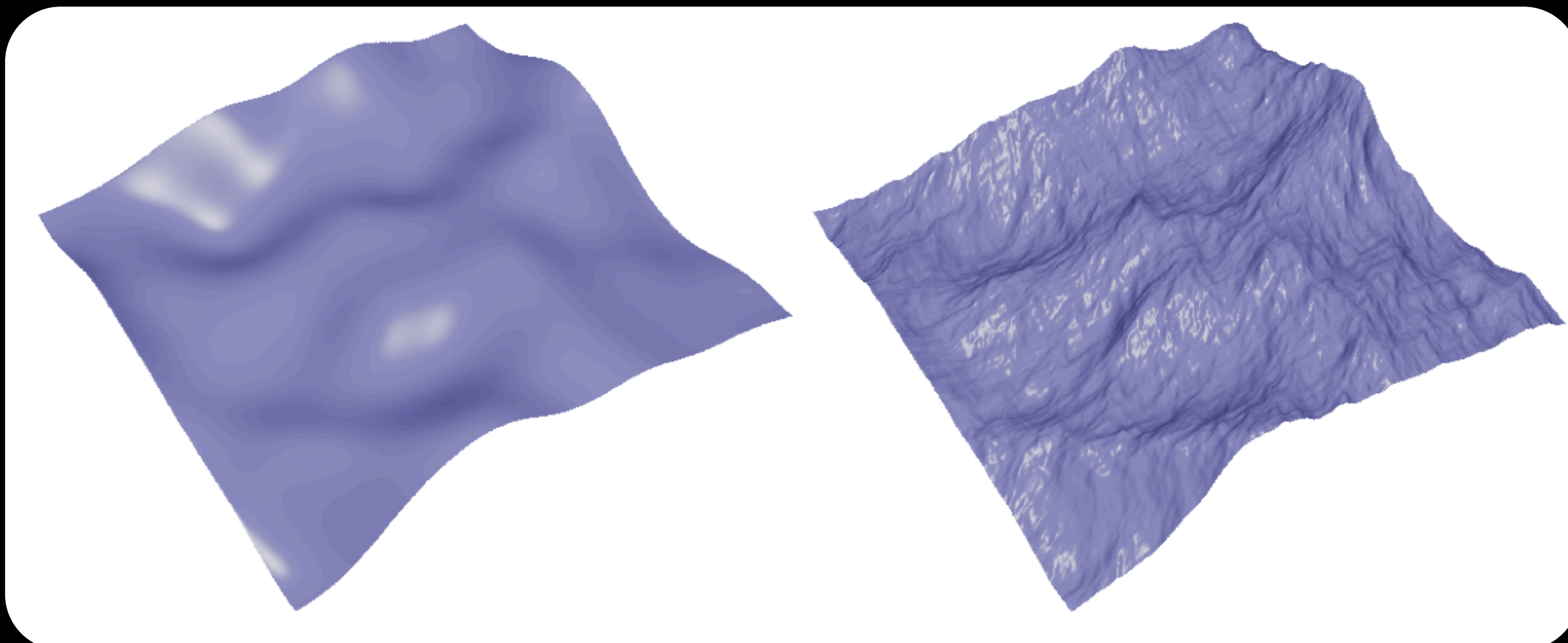
Surface Simulation

[Bojsen-Hansen et al. 2012]



Surface Simulation

[Nielsen et al. 2012]



Outline

- Previous Works
- The Closest Point Method
- 3D iWave
- Additional Extensions
- Results

Closest Point Method

- A simple *embedding* method
  - [Ruuth and Merriman, 2008]
  - [Macdonald and Ruuth 2008]
  - [Macdonald and Ruuth 2009]
  - [Macdonald, Brandman and Ruuth 2011]

2D Wave Equation

$$\frac{\partial^2 \phi}{\partial t^2} = c\nabla^2 \phi$$

2D Wave Equation

$$\boxed{\frac{\partial^2 \phi}{\partial t^2}} = c\nabla^2\phi$$

2D Wave Equation

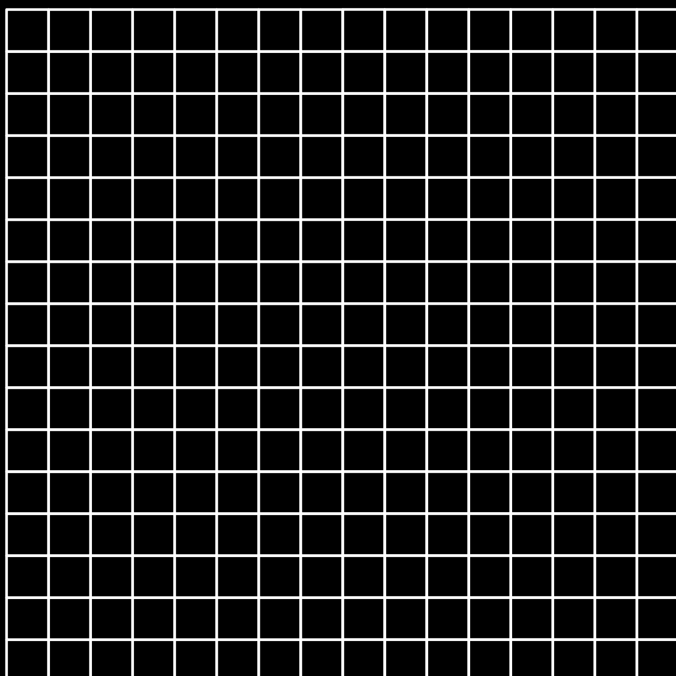
$$\frac{\partial^2 \phi}{\partial t^2} = \boxed{c \nabla^2 \phi}$$

2D Wave Equation

$$\frac{\partial^2 \phi}{\partial t^2} = \boxed{c \nabla^2 \phi} \Rightarrow$$

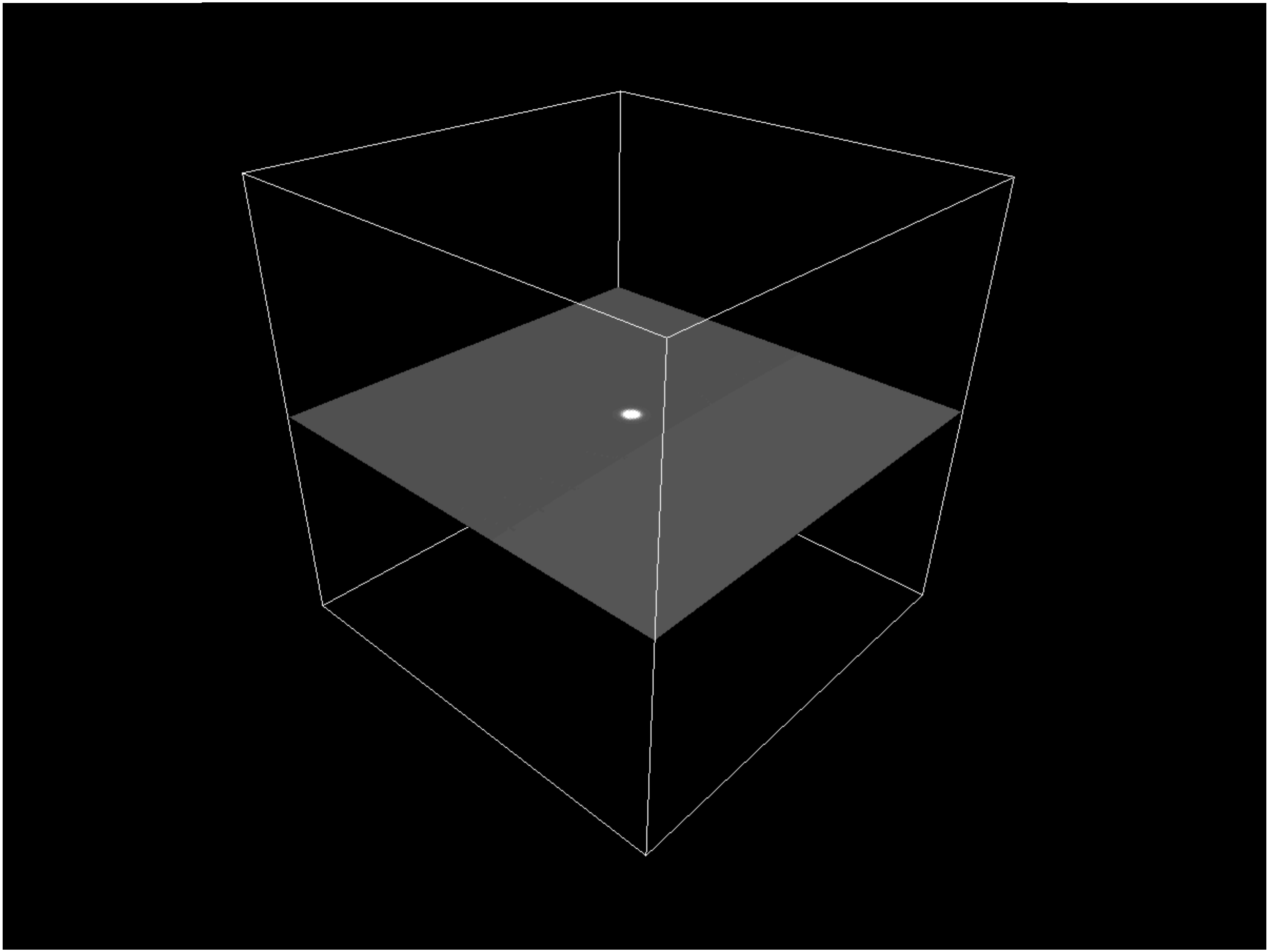
	1	
1	-4	1
	1	

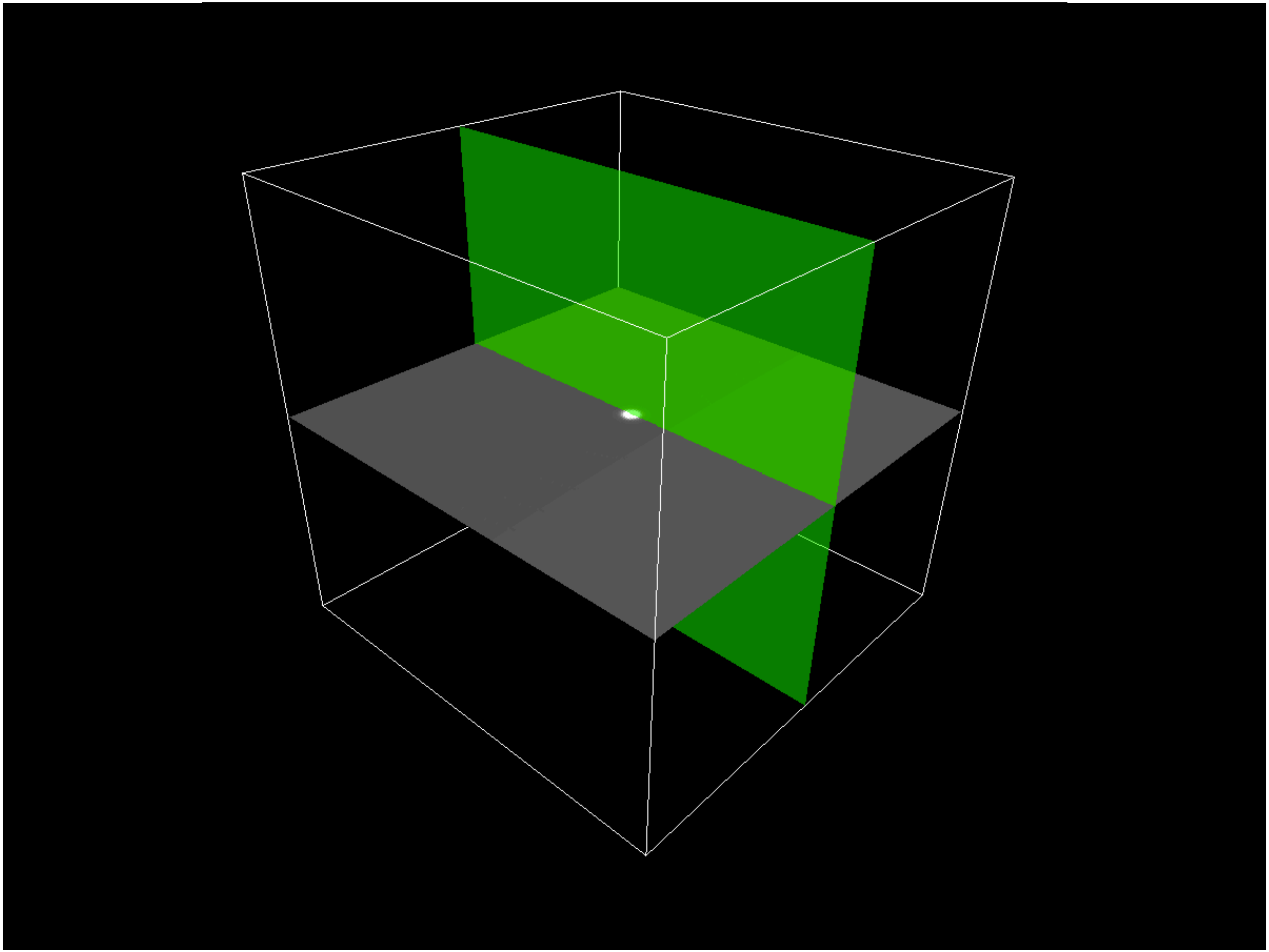
2D Wave Equation

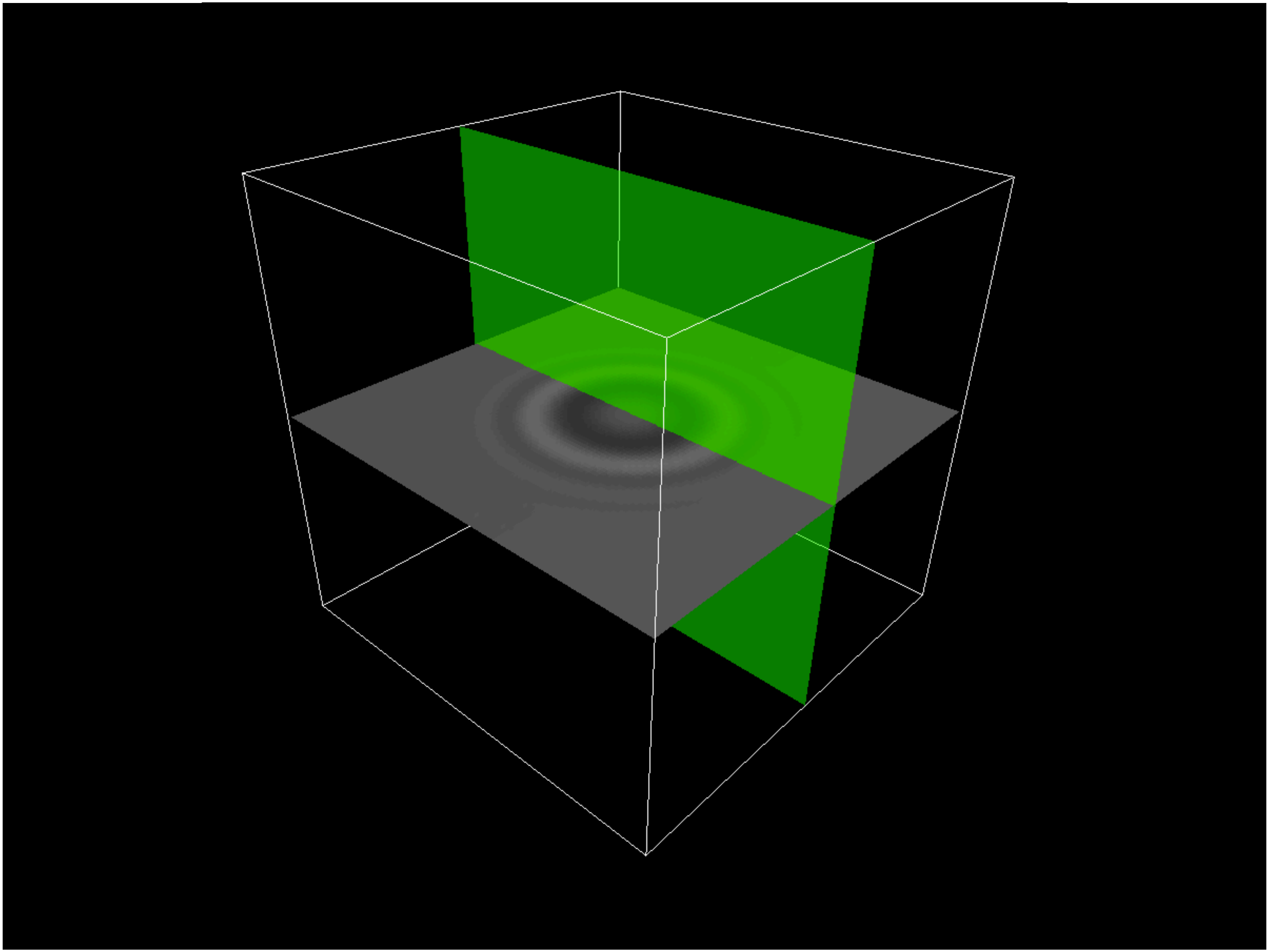


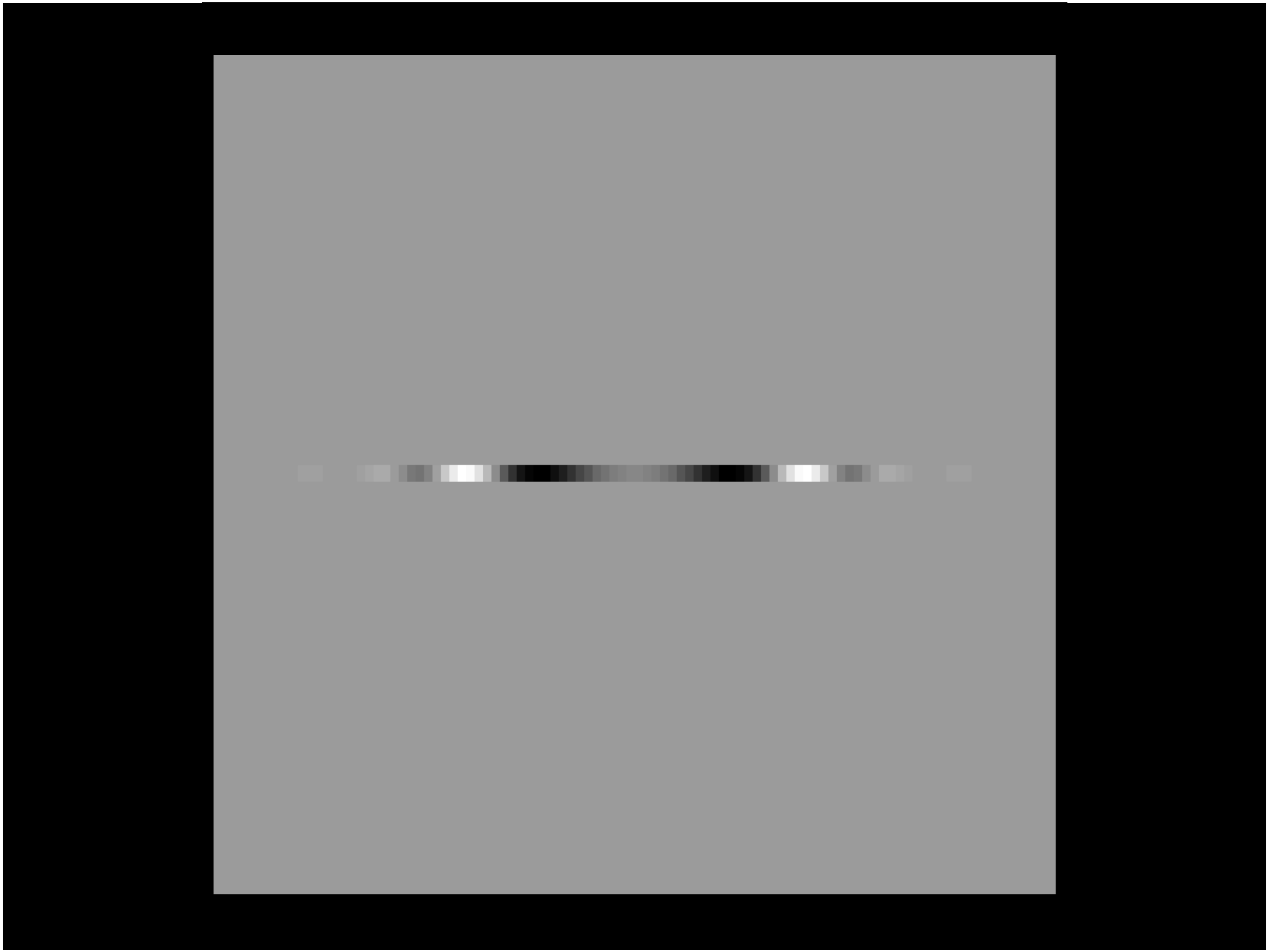
ϕ

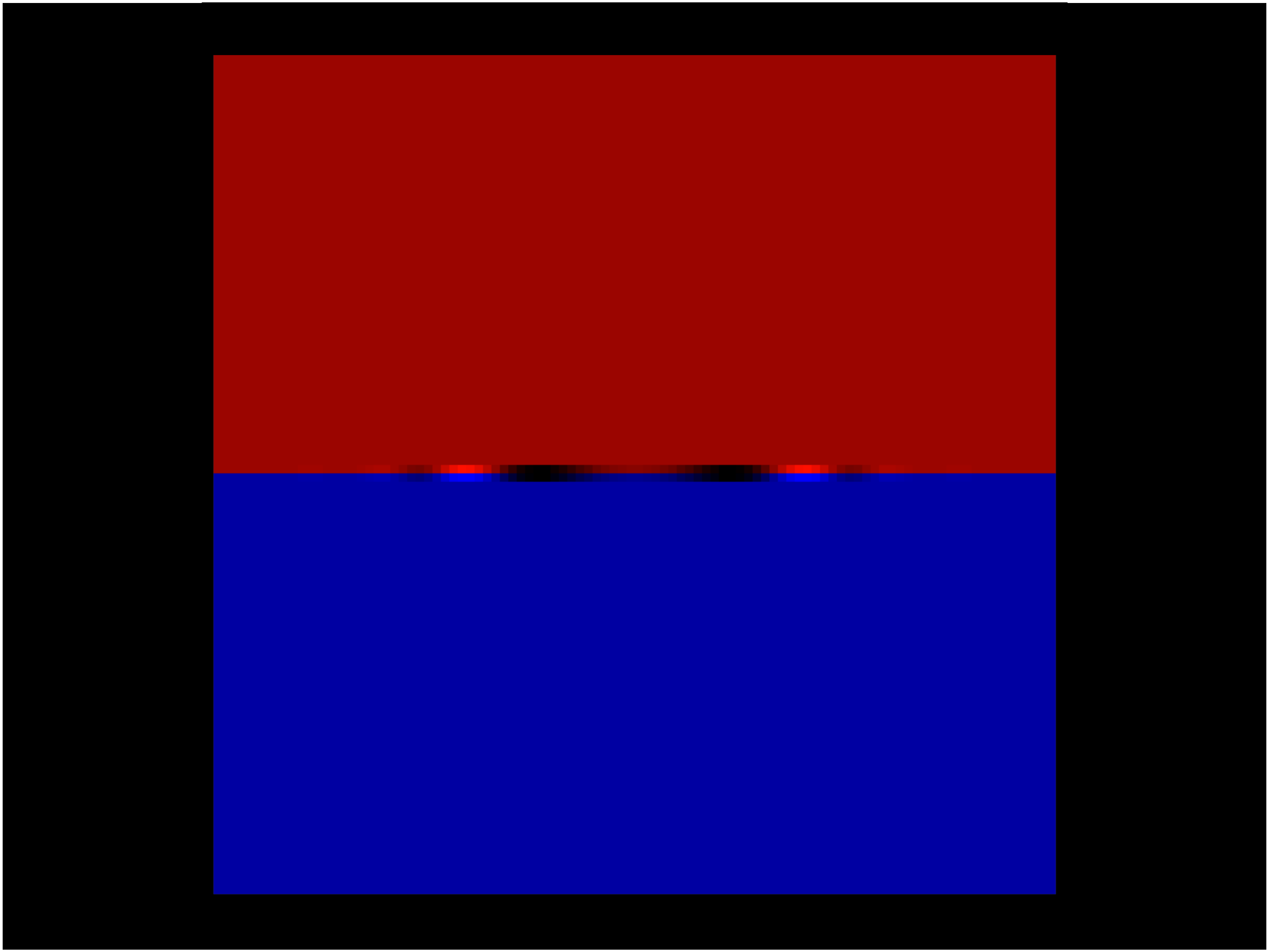
	1	
1	-4	1
	1	

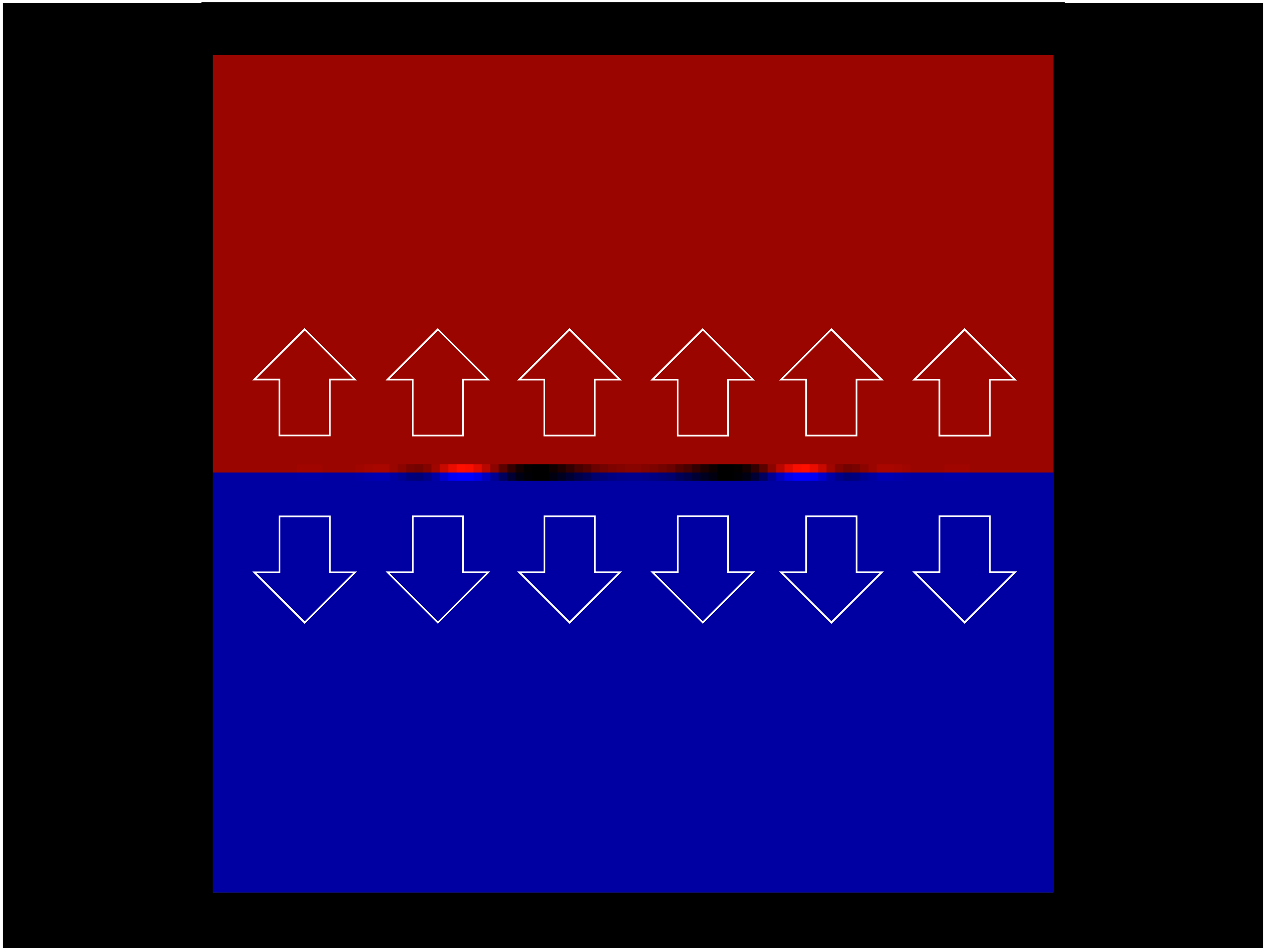


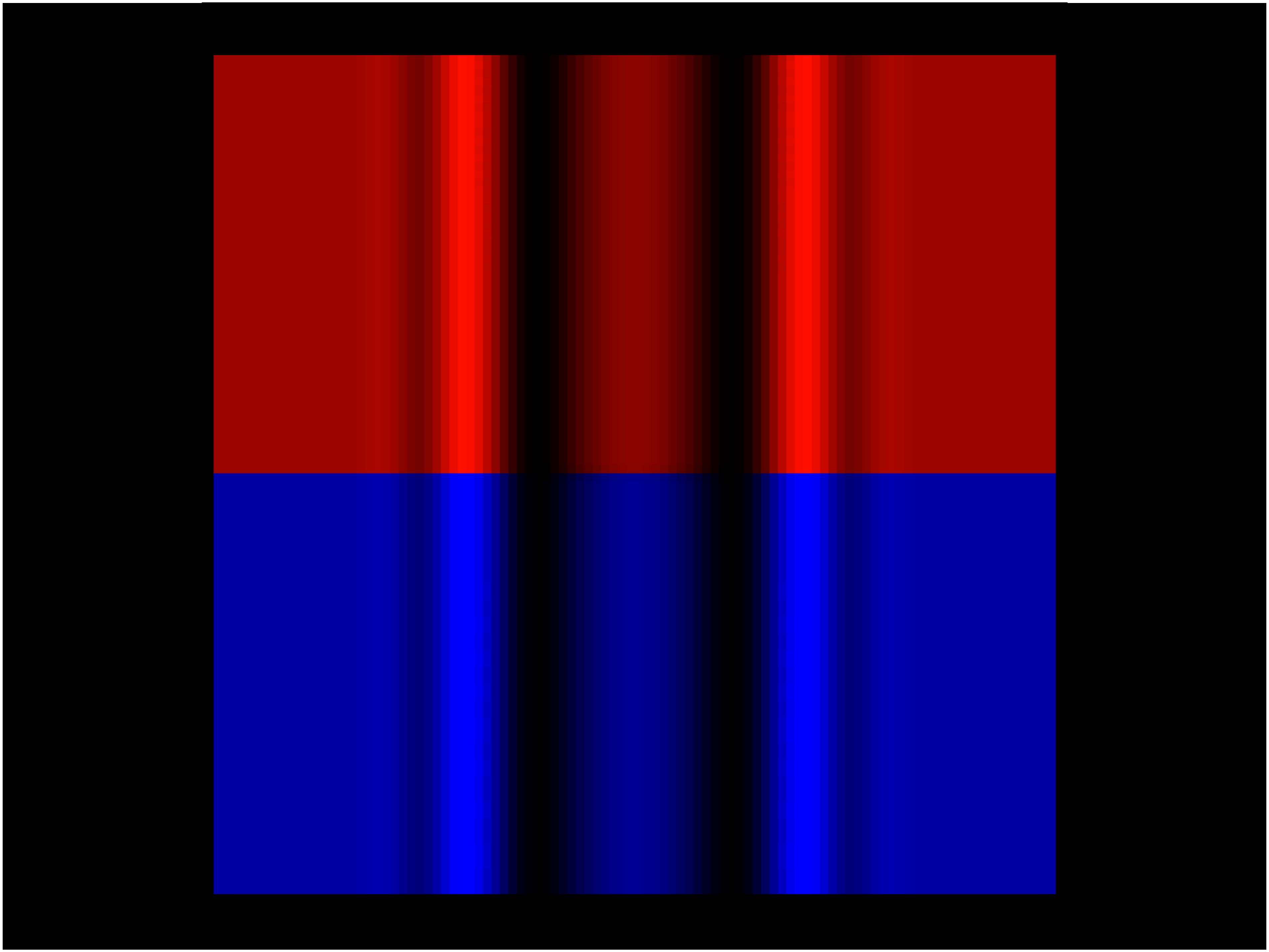


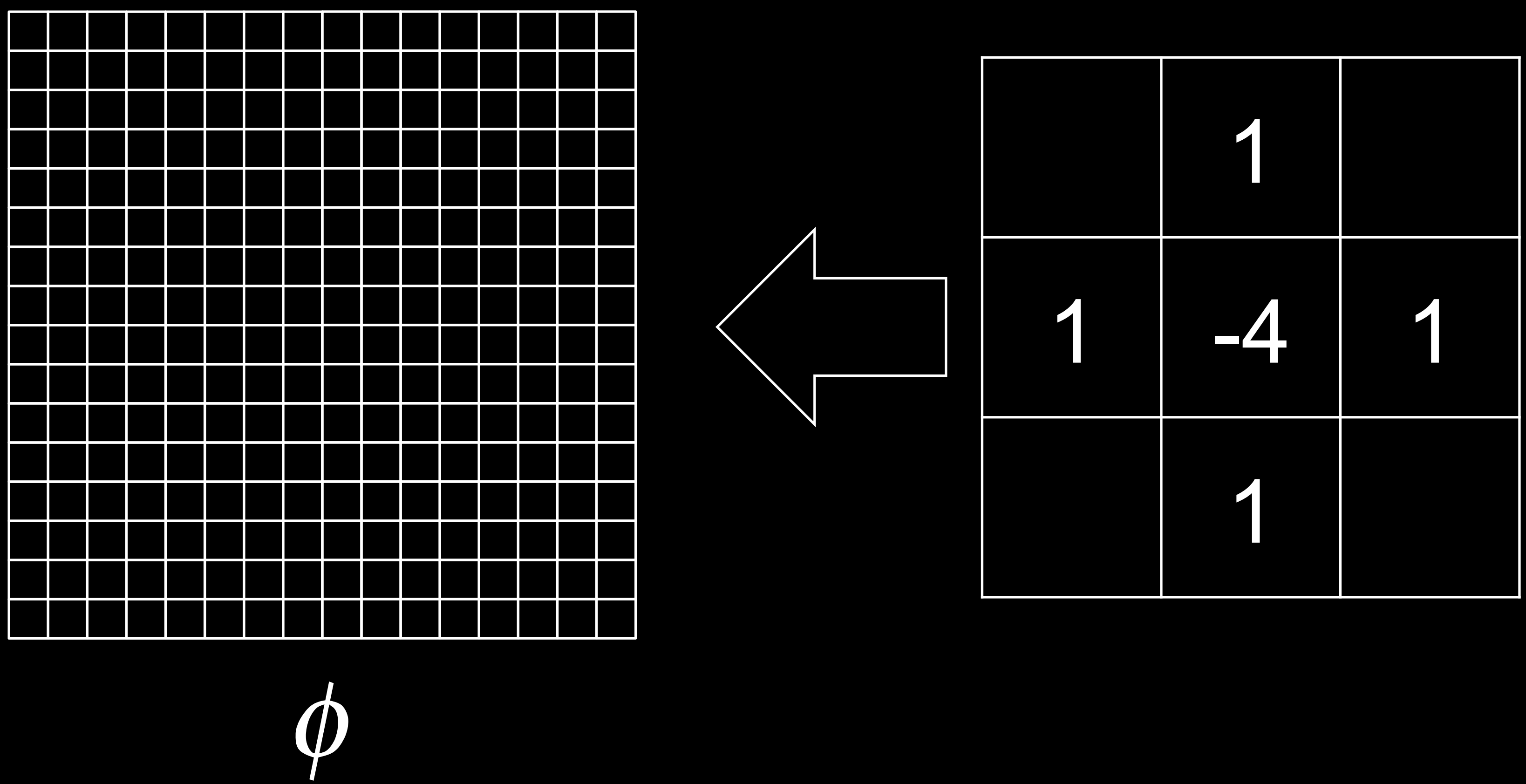


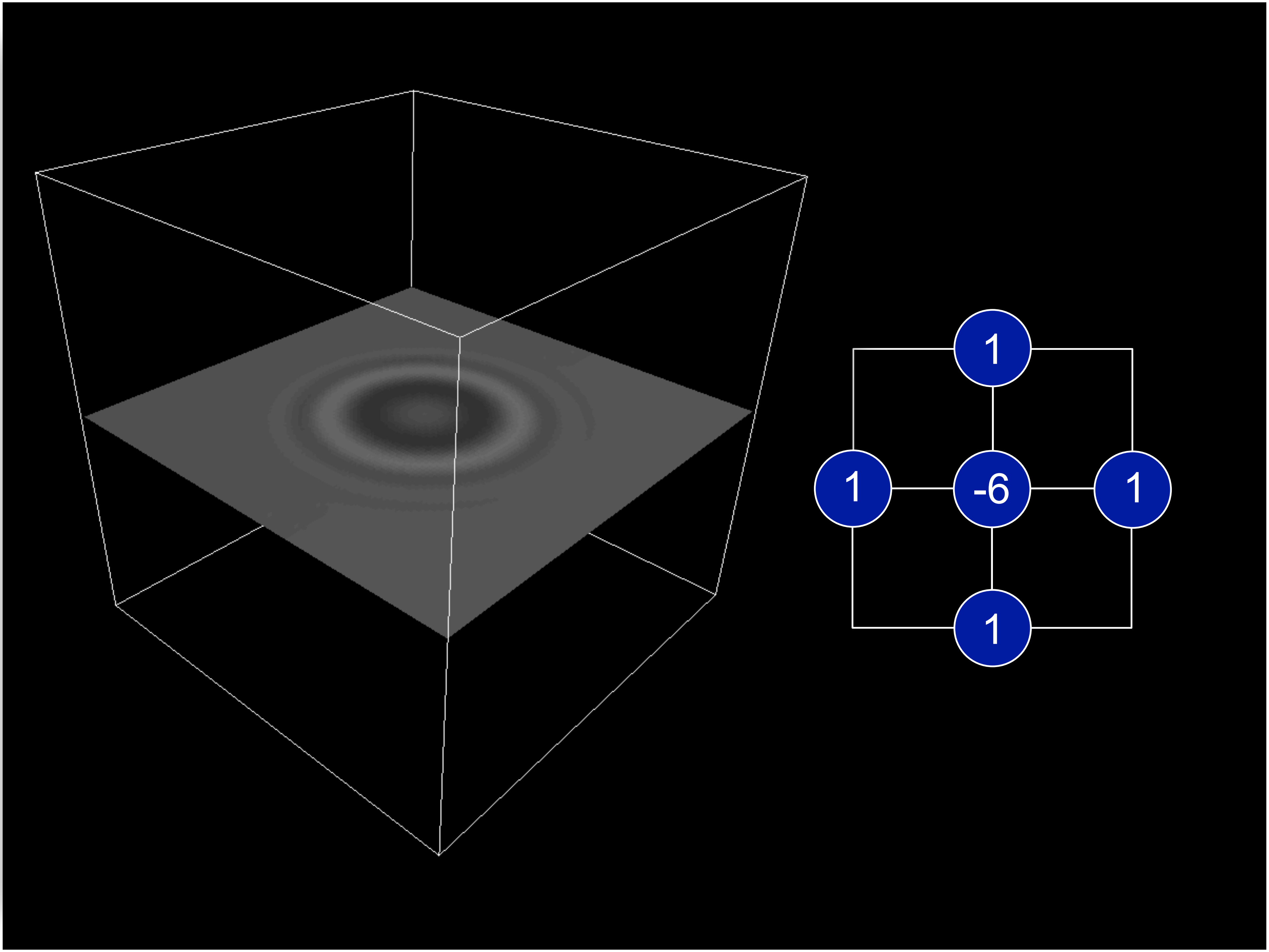


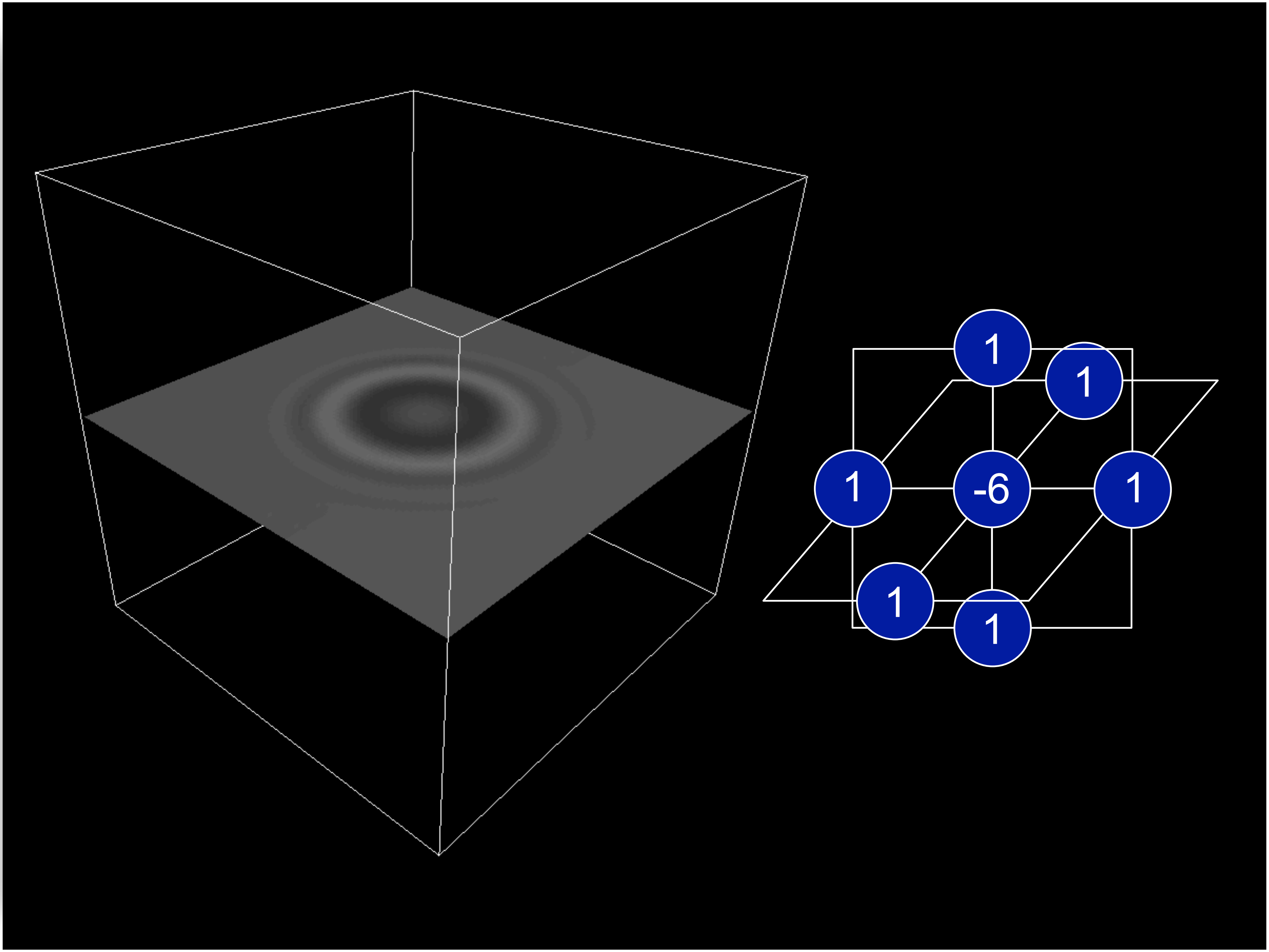


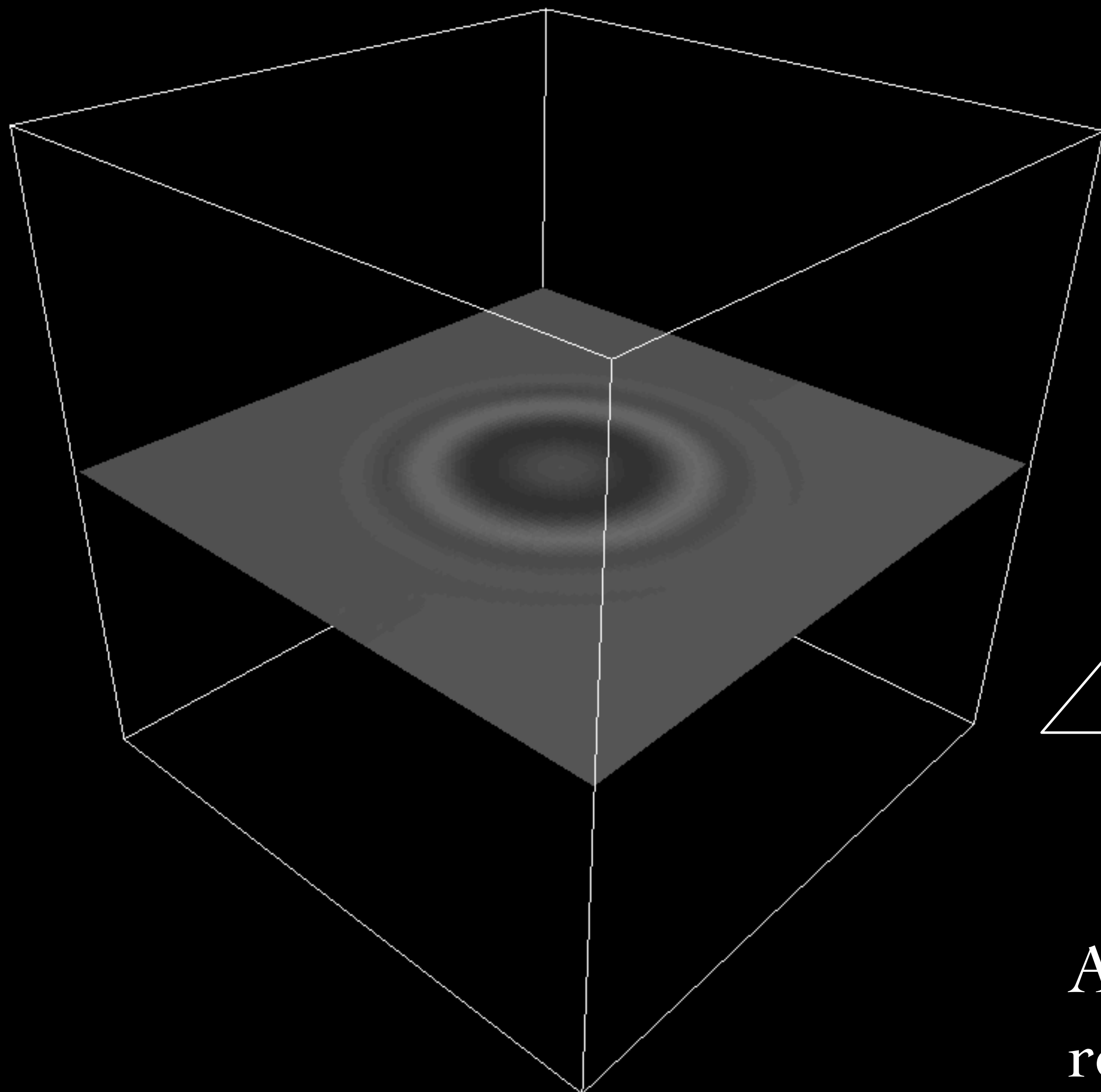


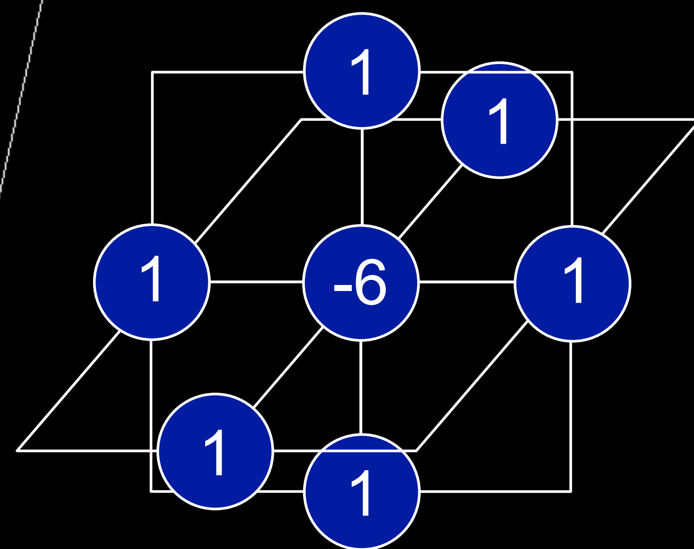



1
1 -4 1
1
ϕ

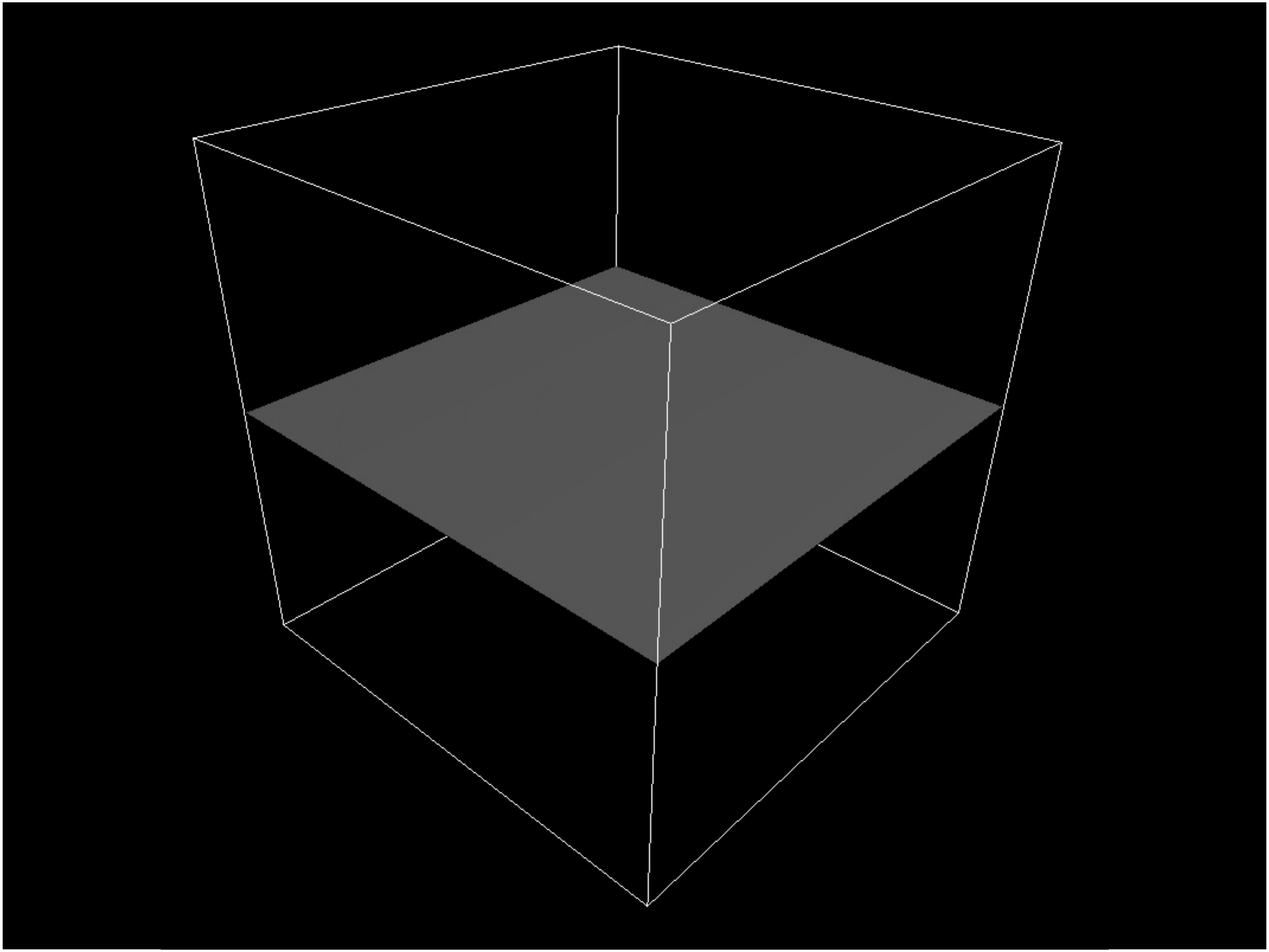

1
1
-6
1
1

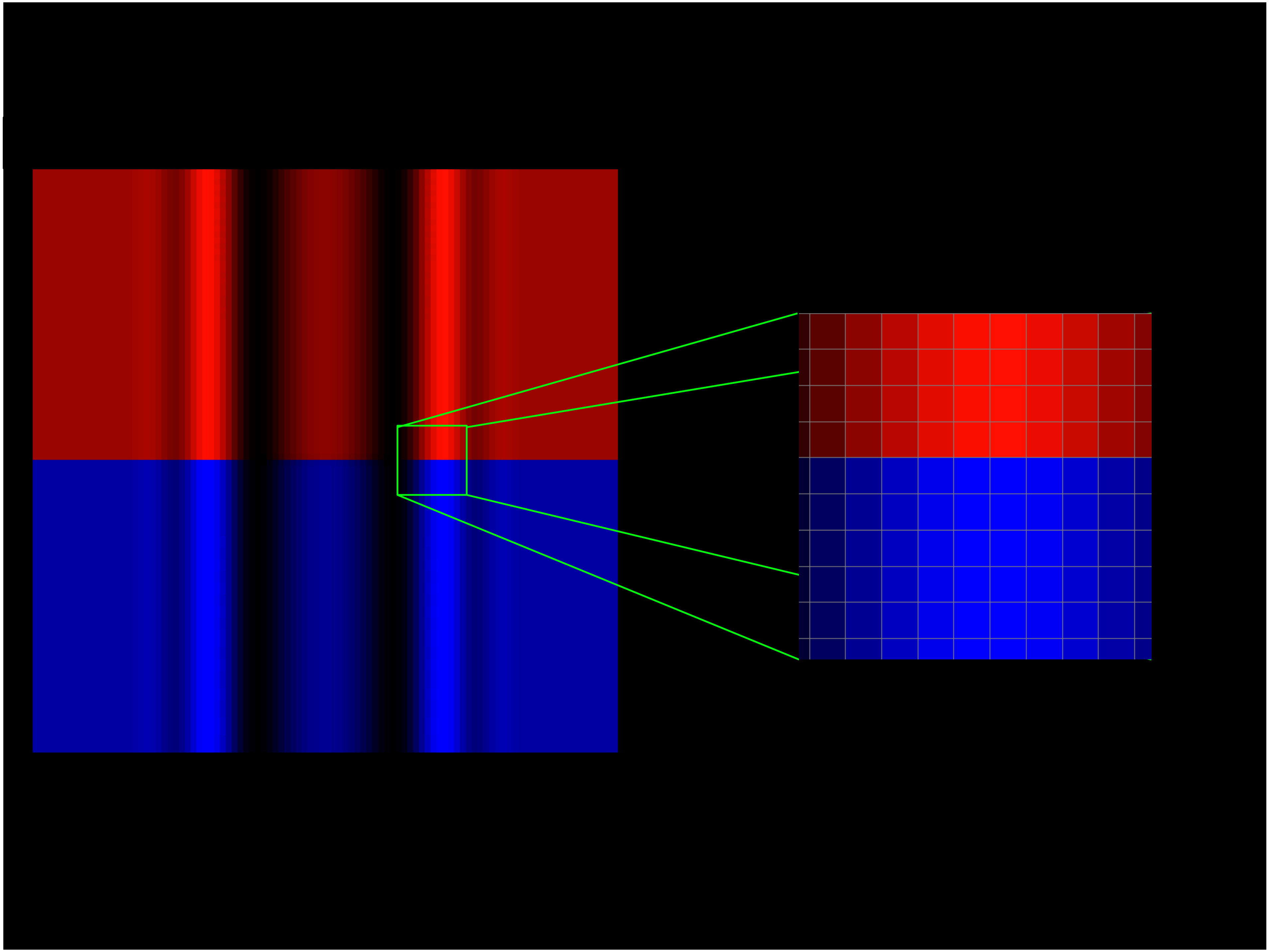

1
1
1
-6
1
1
1

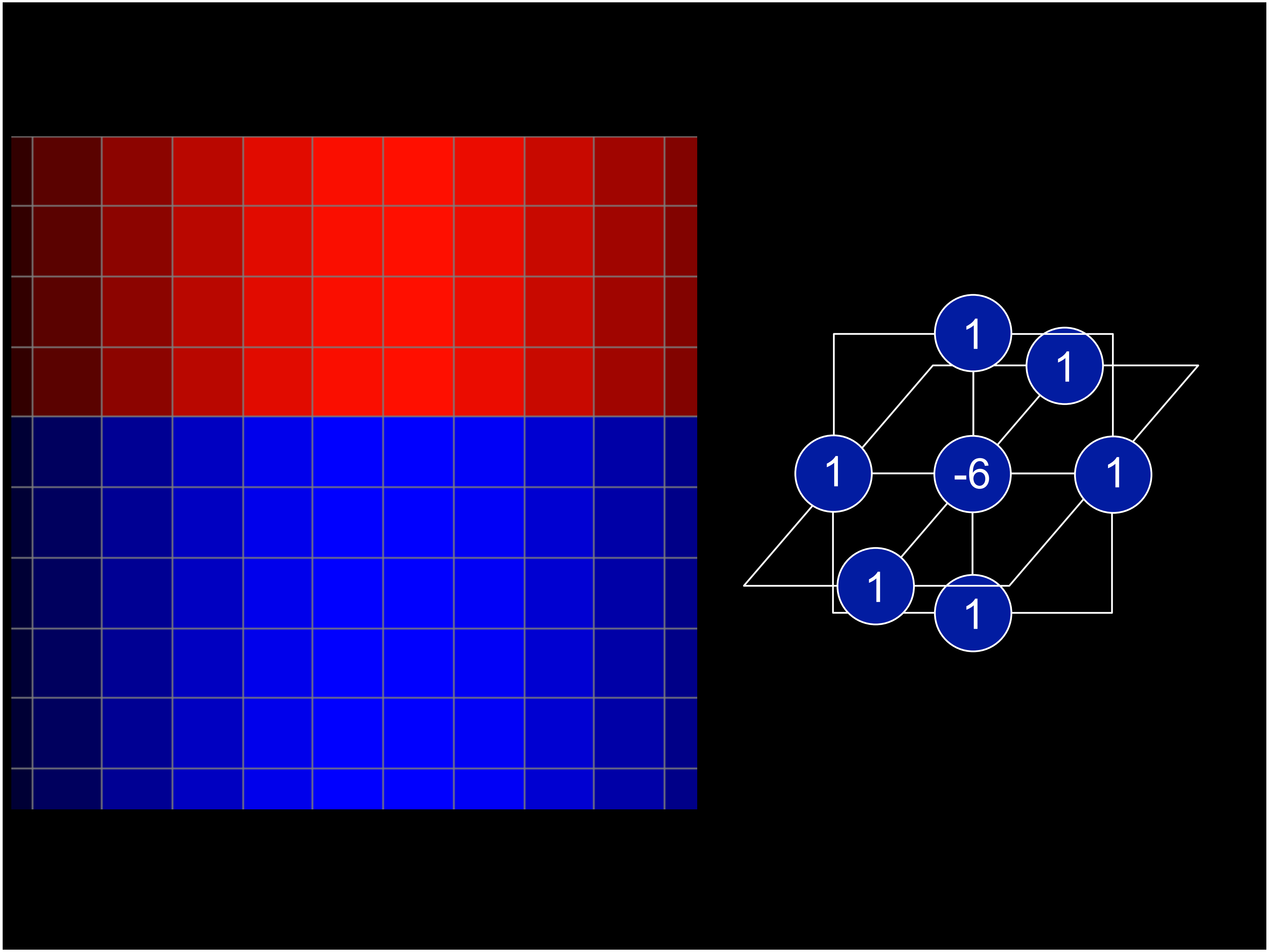




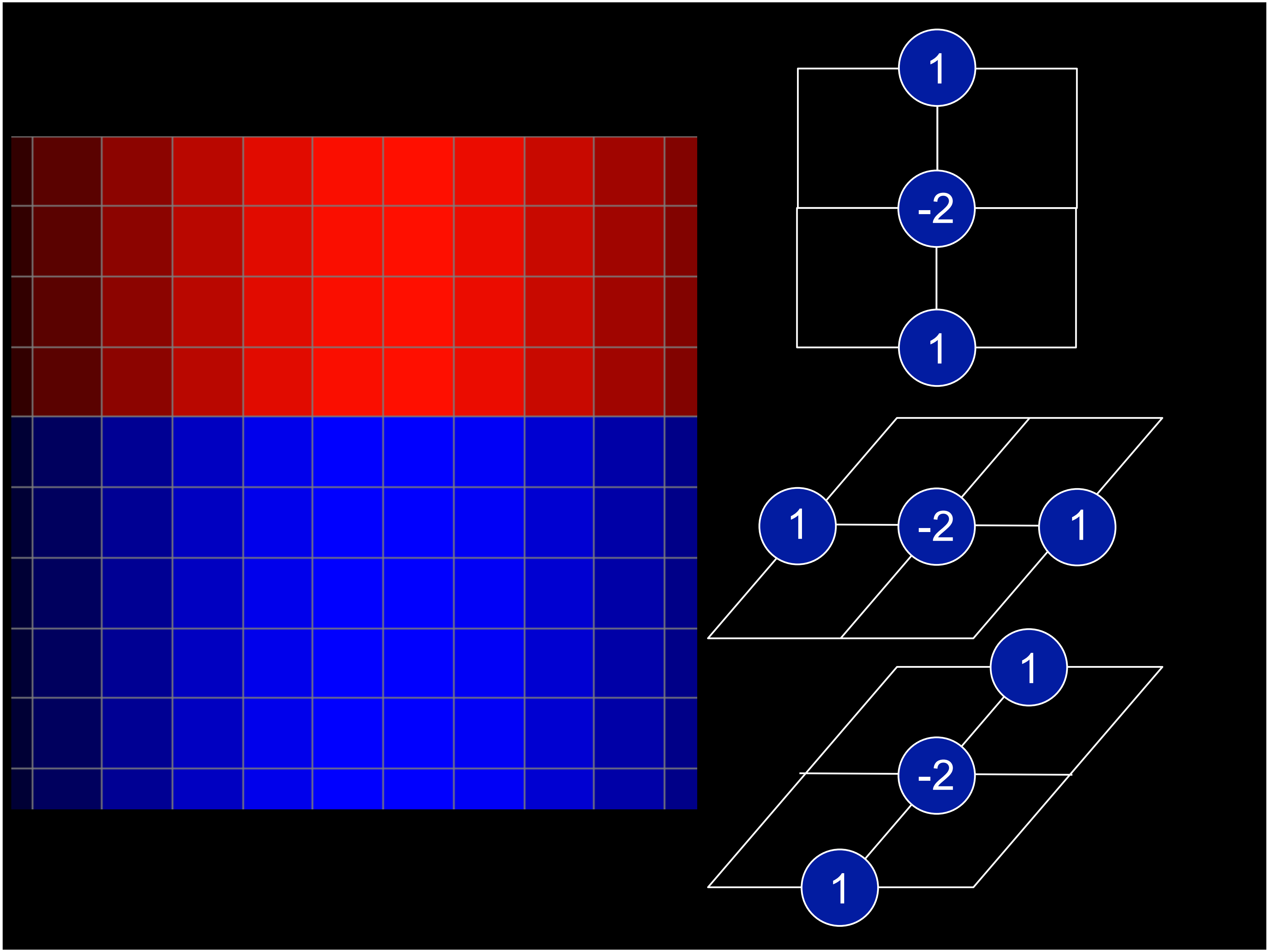
After each step,
re-extend

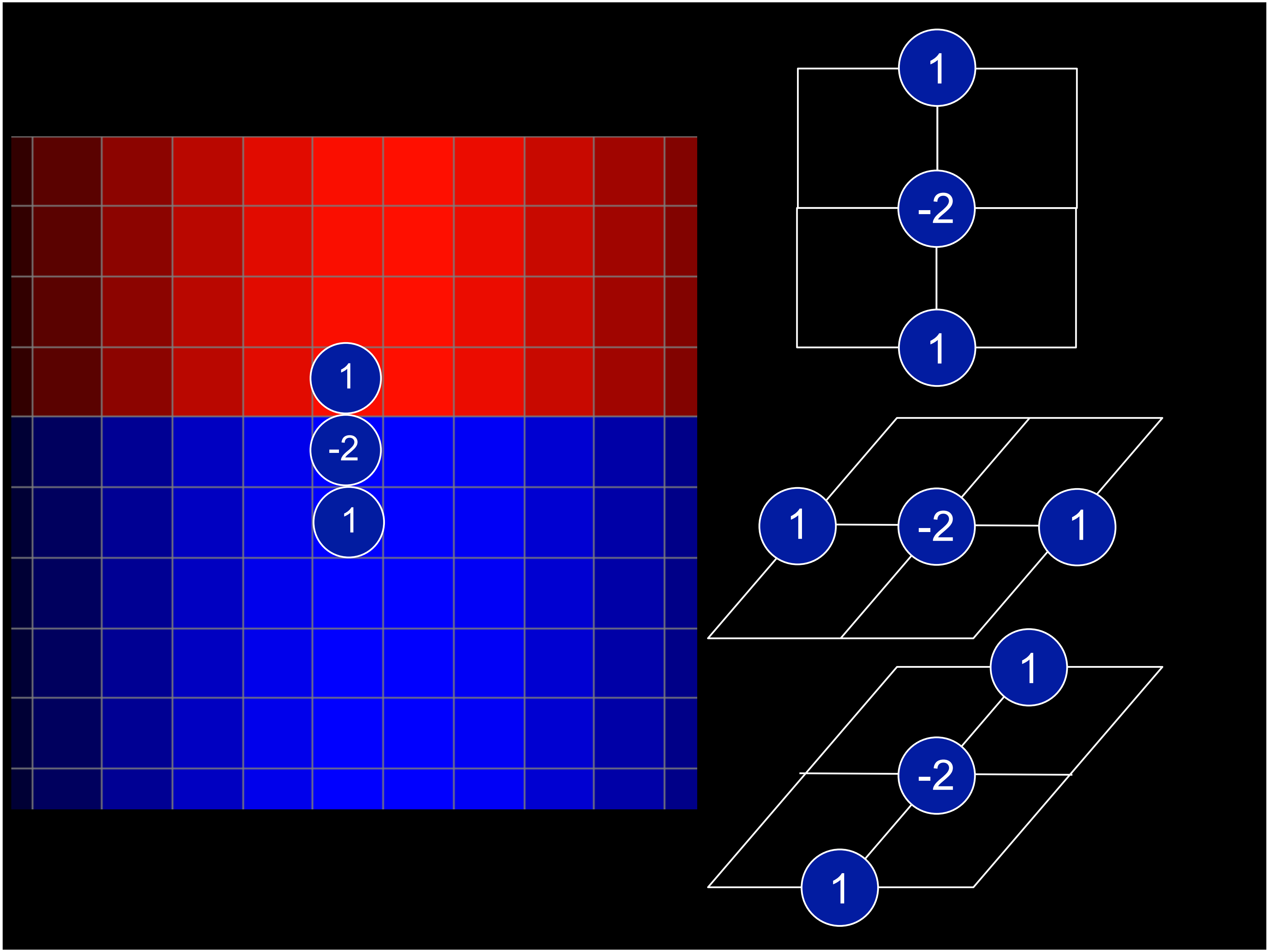




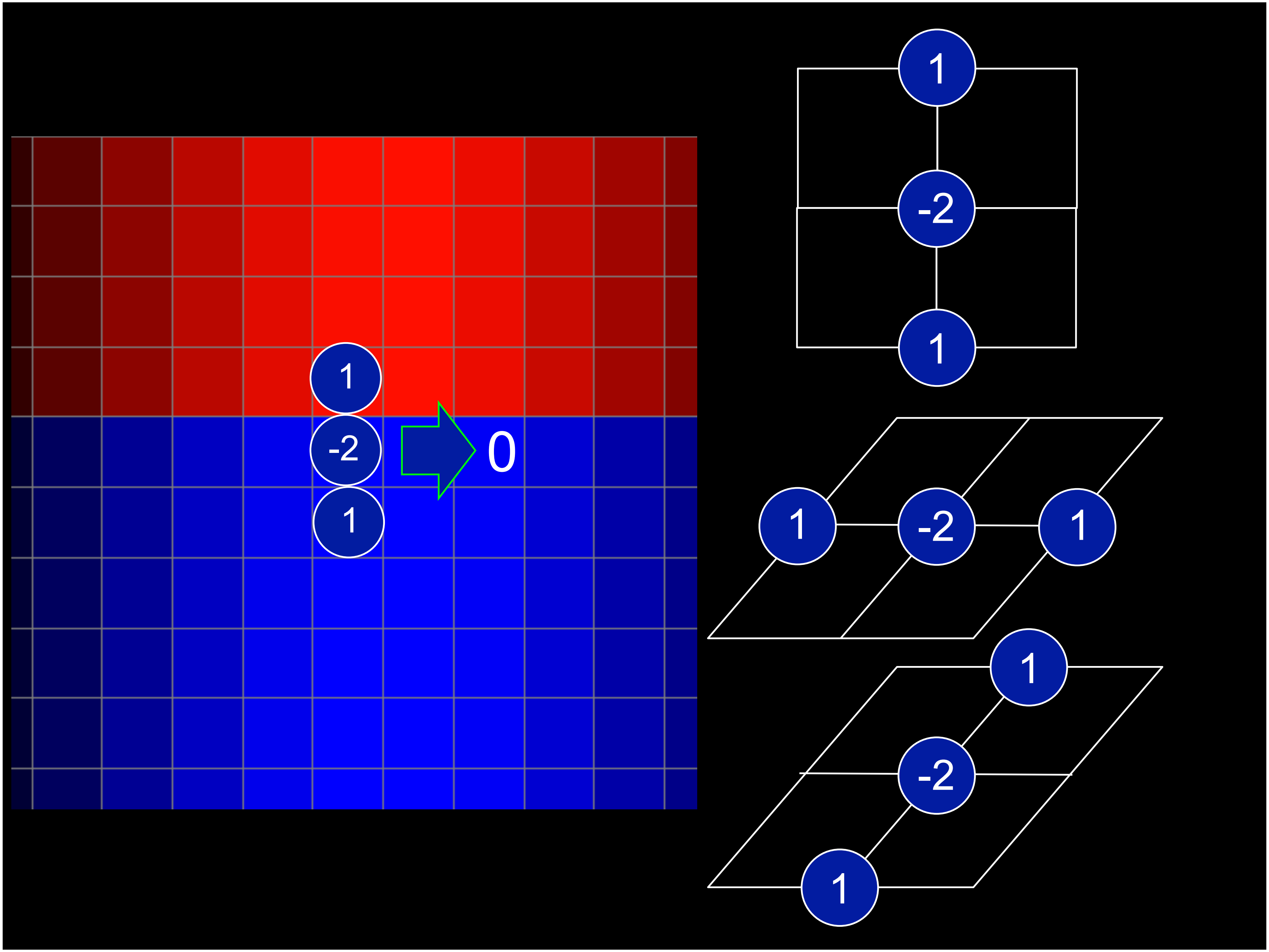


1
1
1
-6
1
1
1

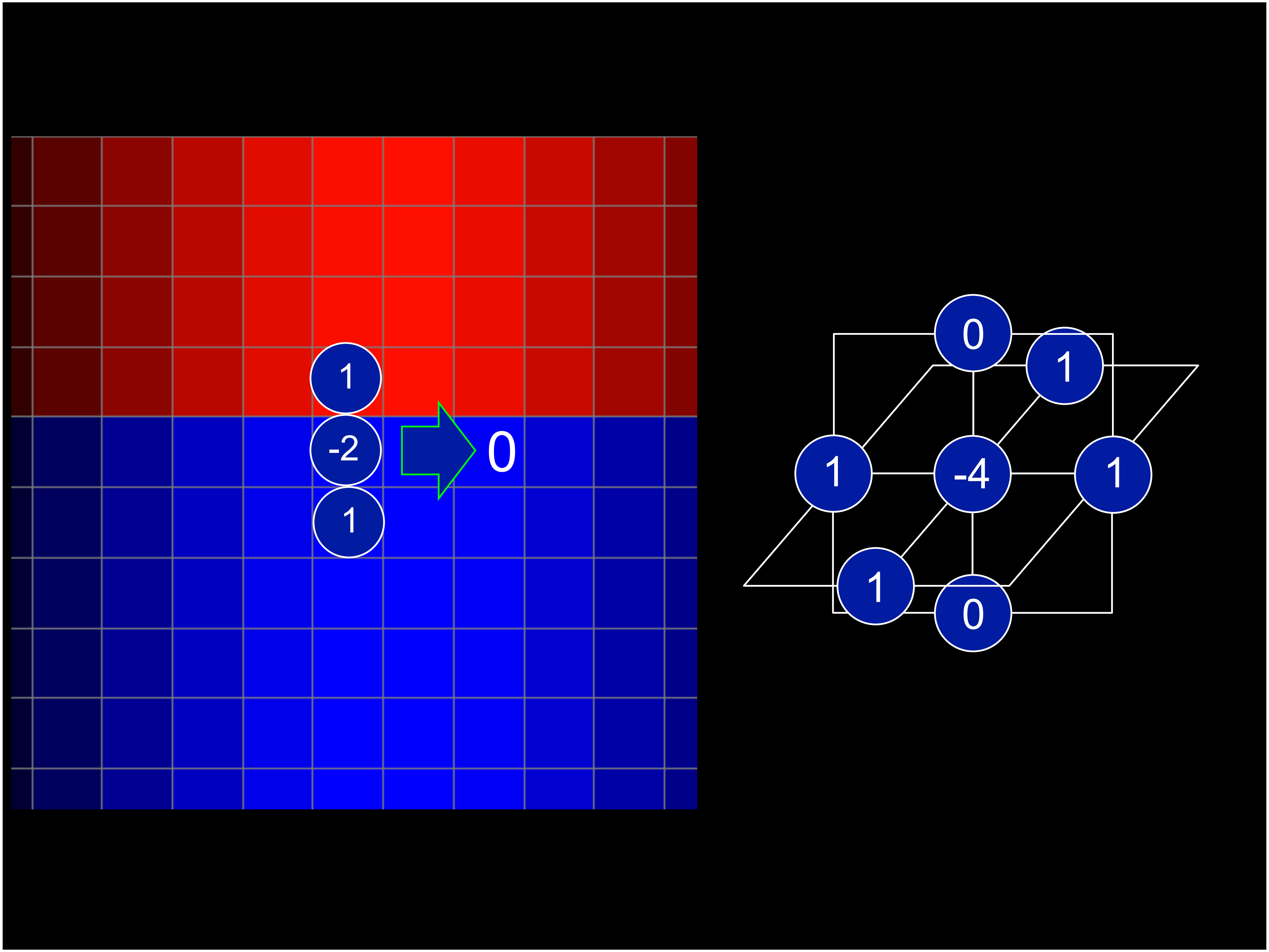

1
-2
1
1
-2
1
1
-2
1



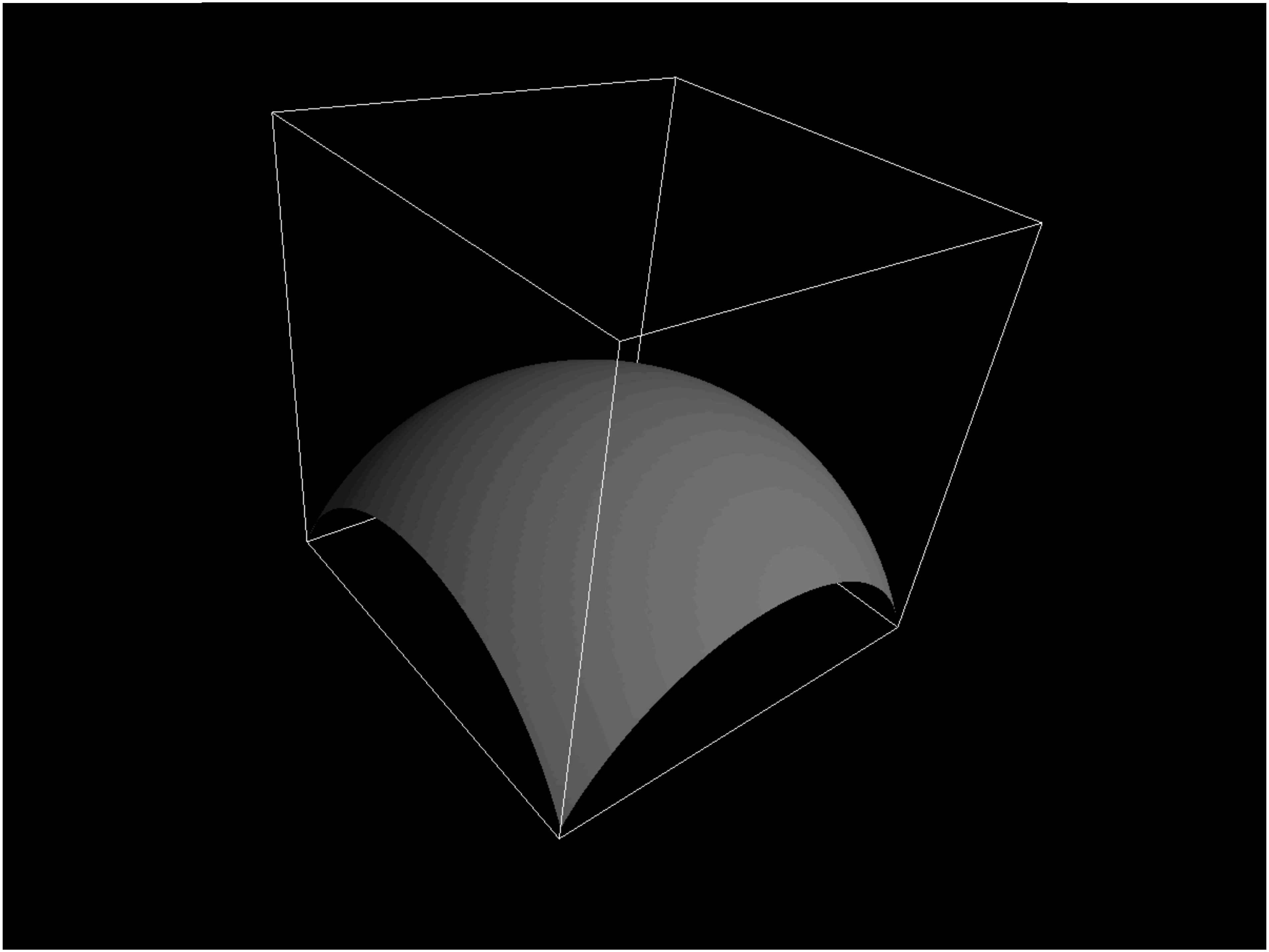
1
-2
1
1
-2
1
1
-2
1
1
-2
1

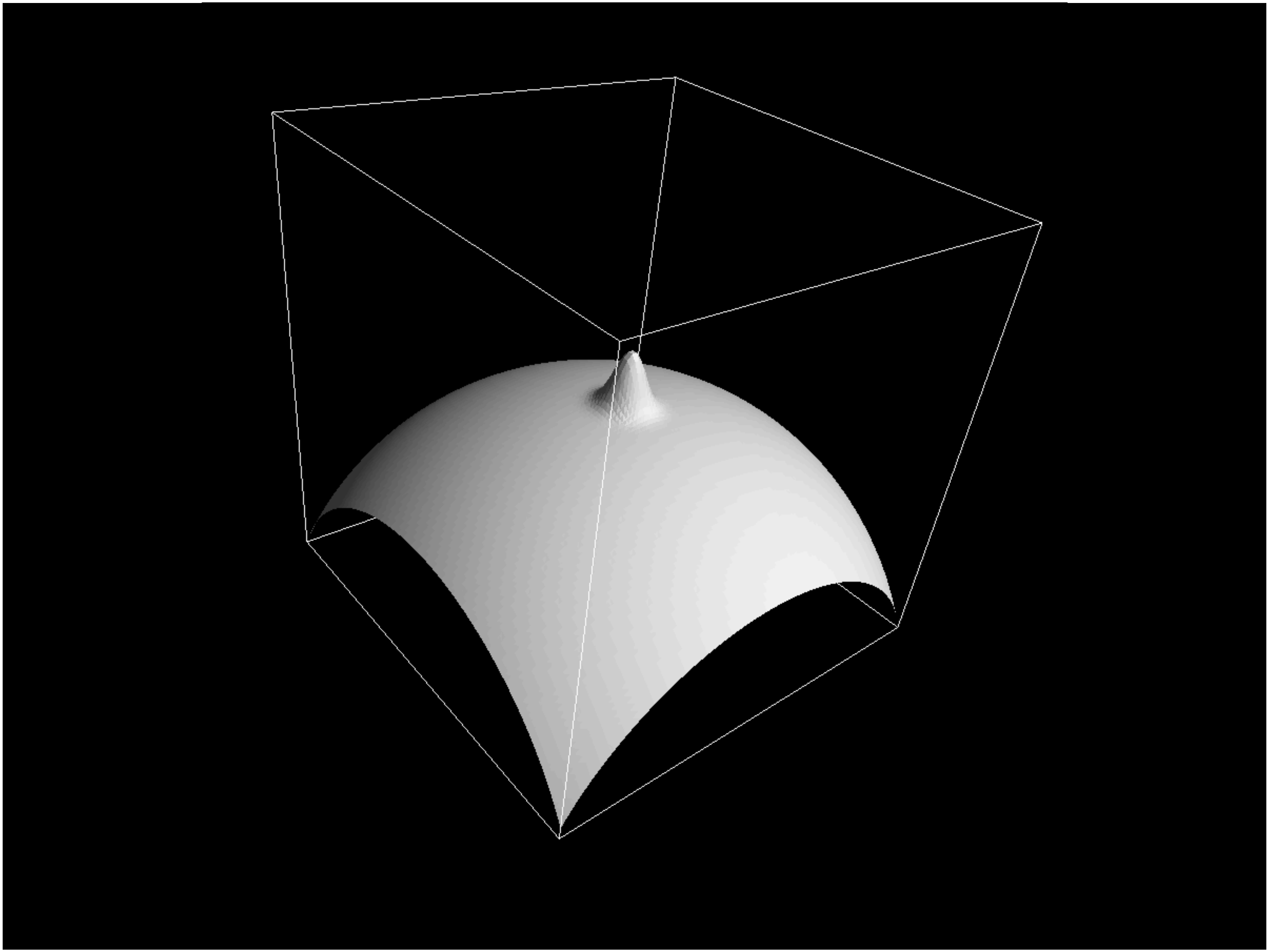


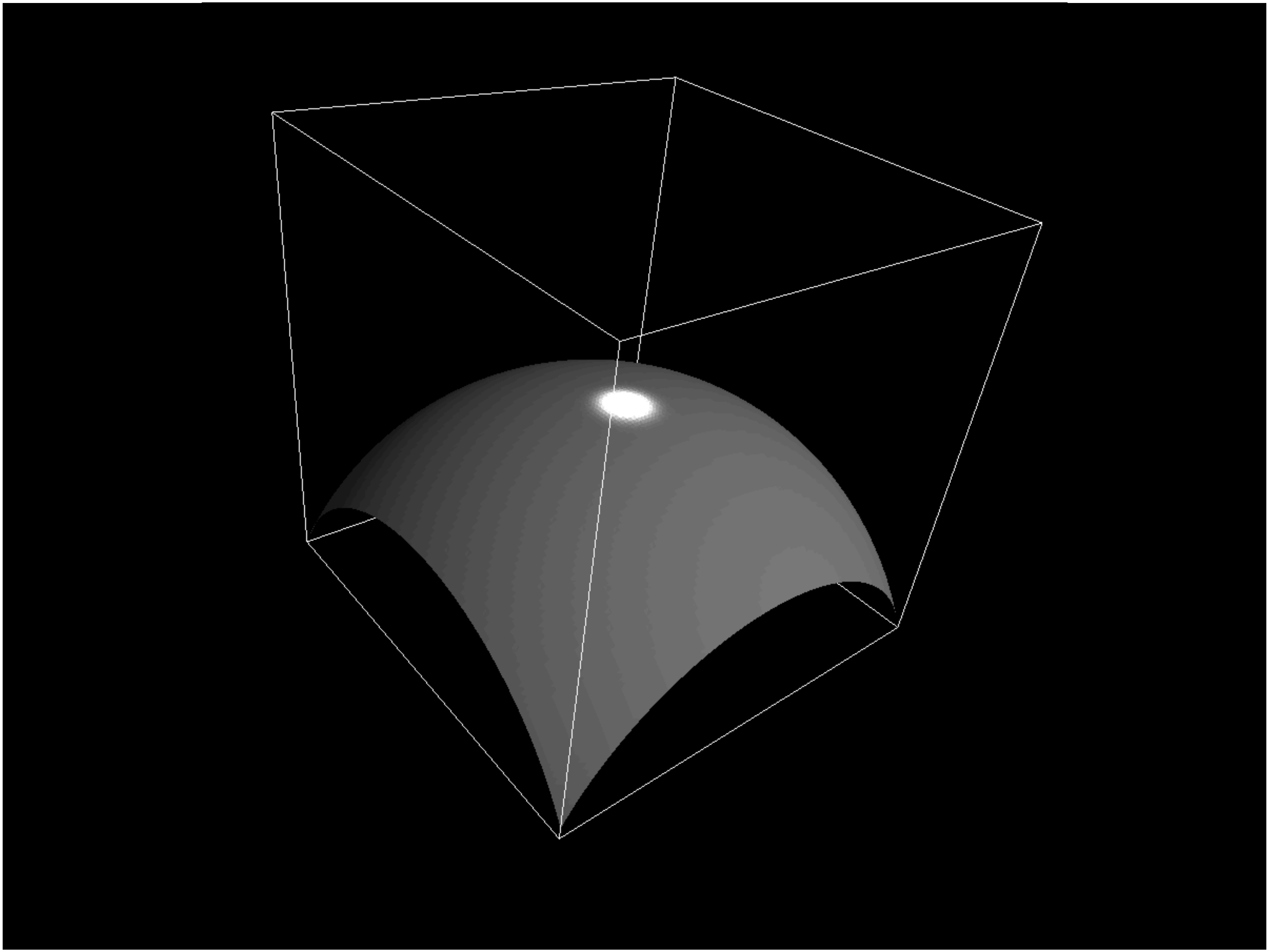
1
-2
1
0
1
-2
1
1
-2
1
1
-2
1

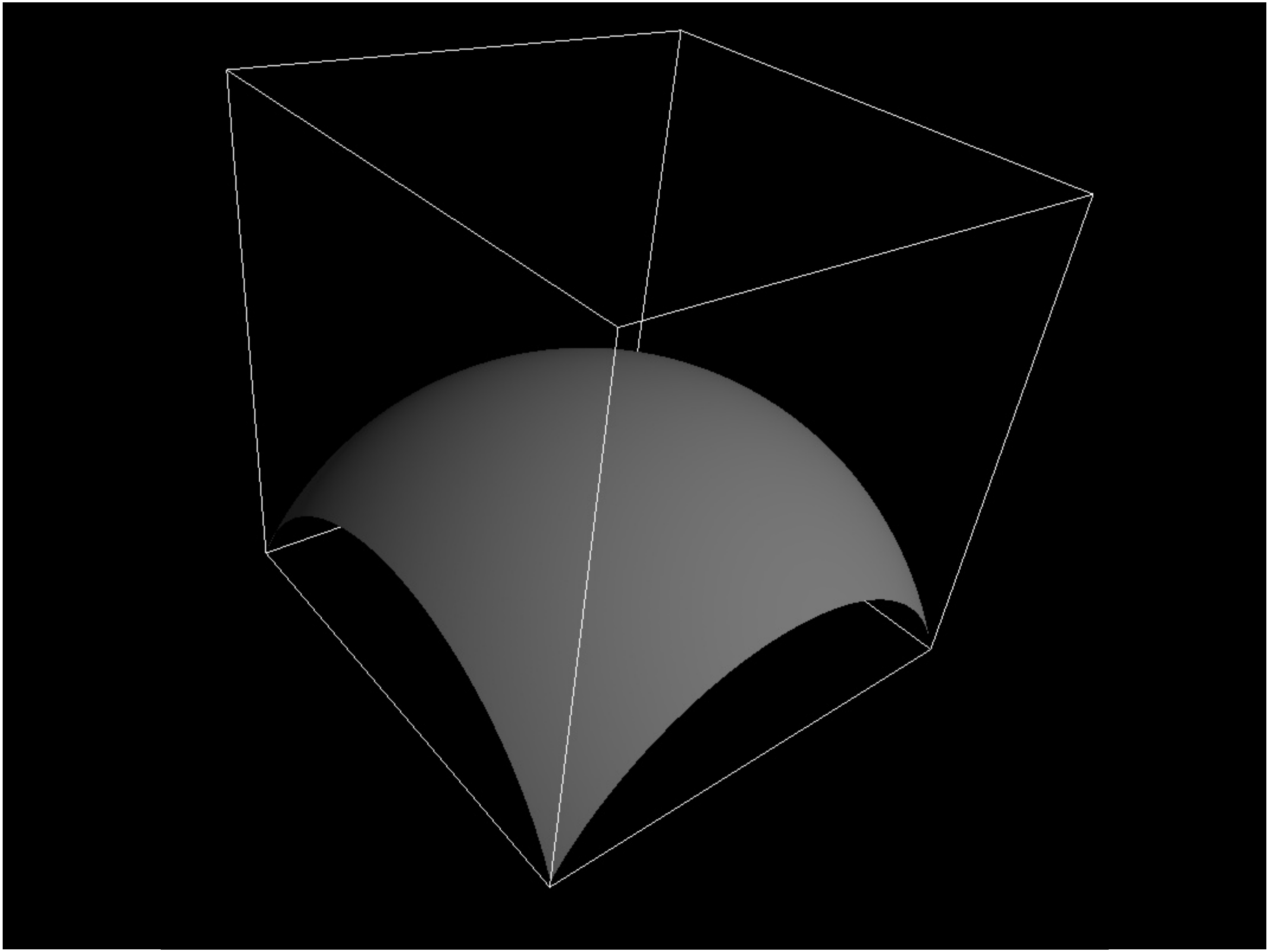


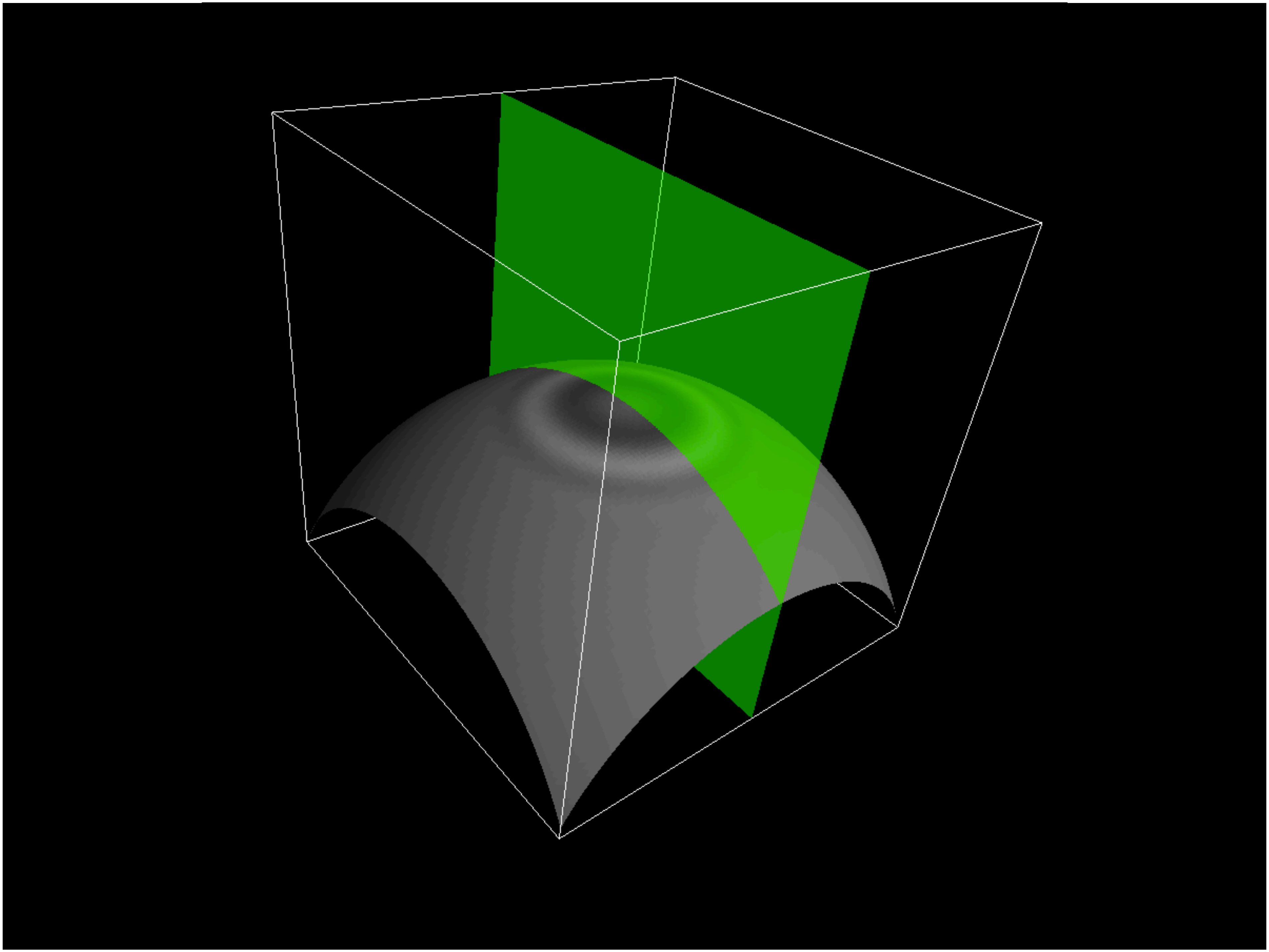
1
-2
1
0
0
1
1
-4
1
1
0

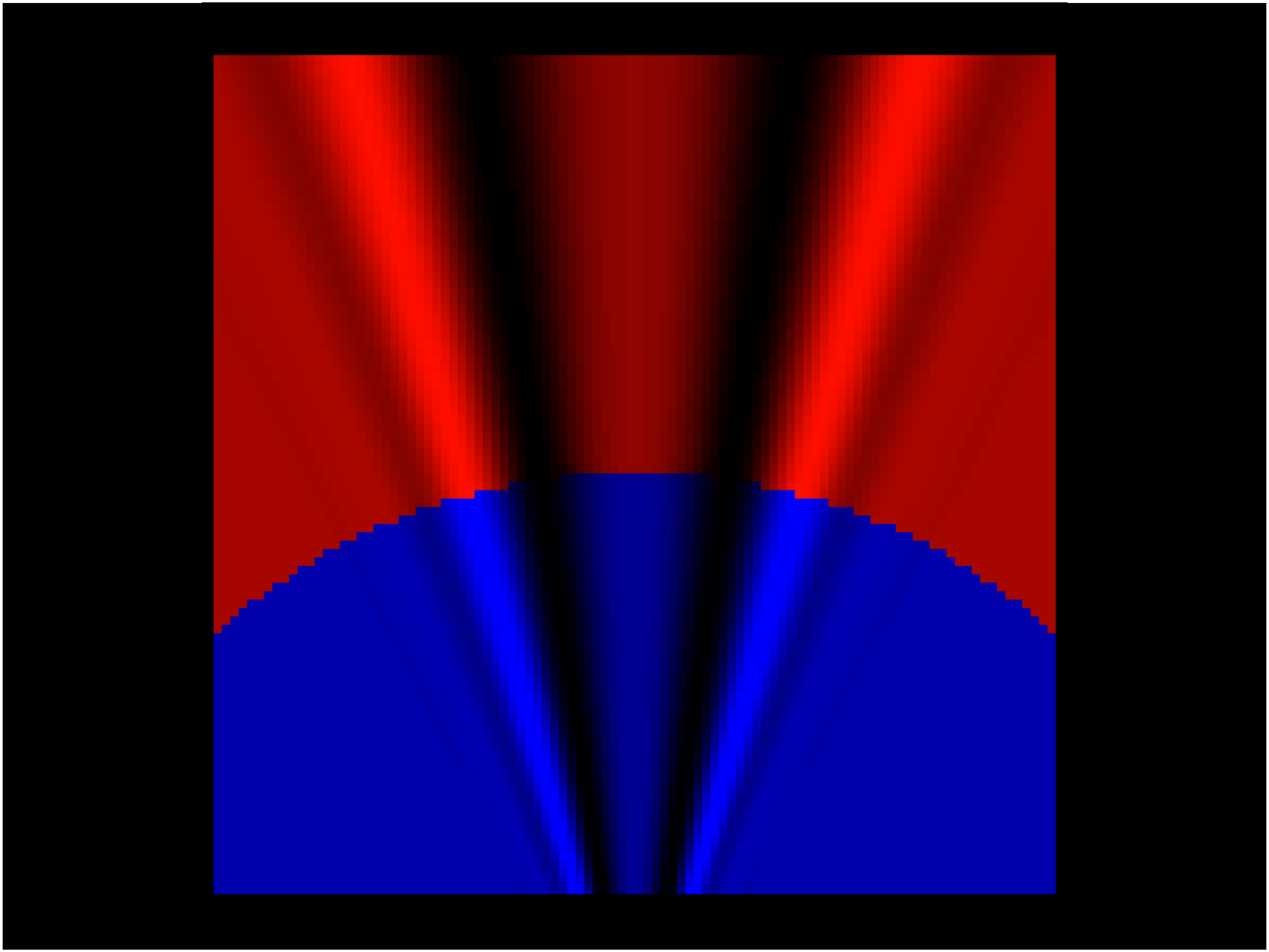












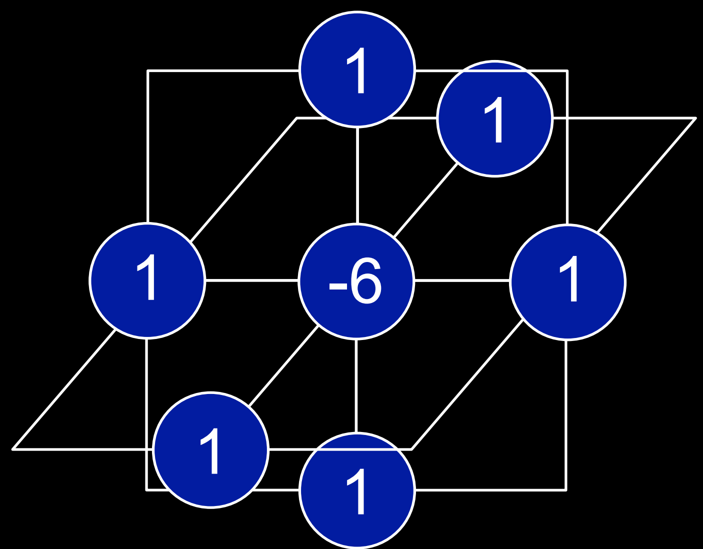
Outline

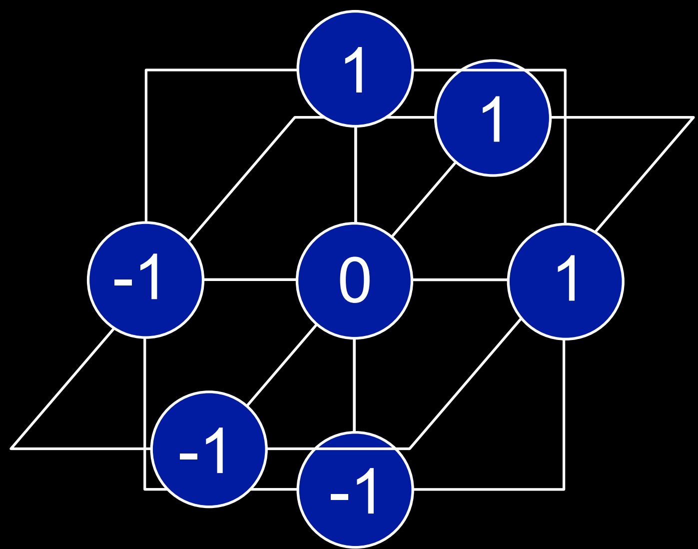
- Previous Works
- The Closest Point Method
- 3D iWave
- Additional Extensions
- Results

Closest Point Method

- Works well for

Laplacian: $\nabla^2\phi$ Gradient: $\nabla \cdot \phi$



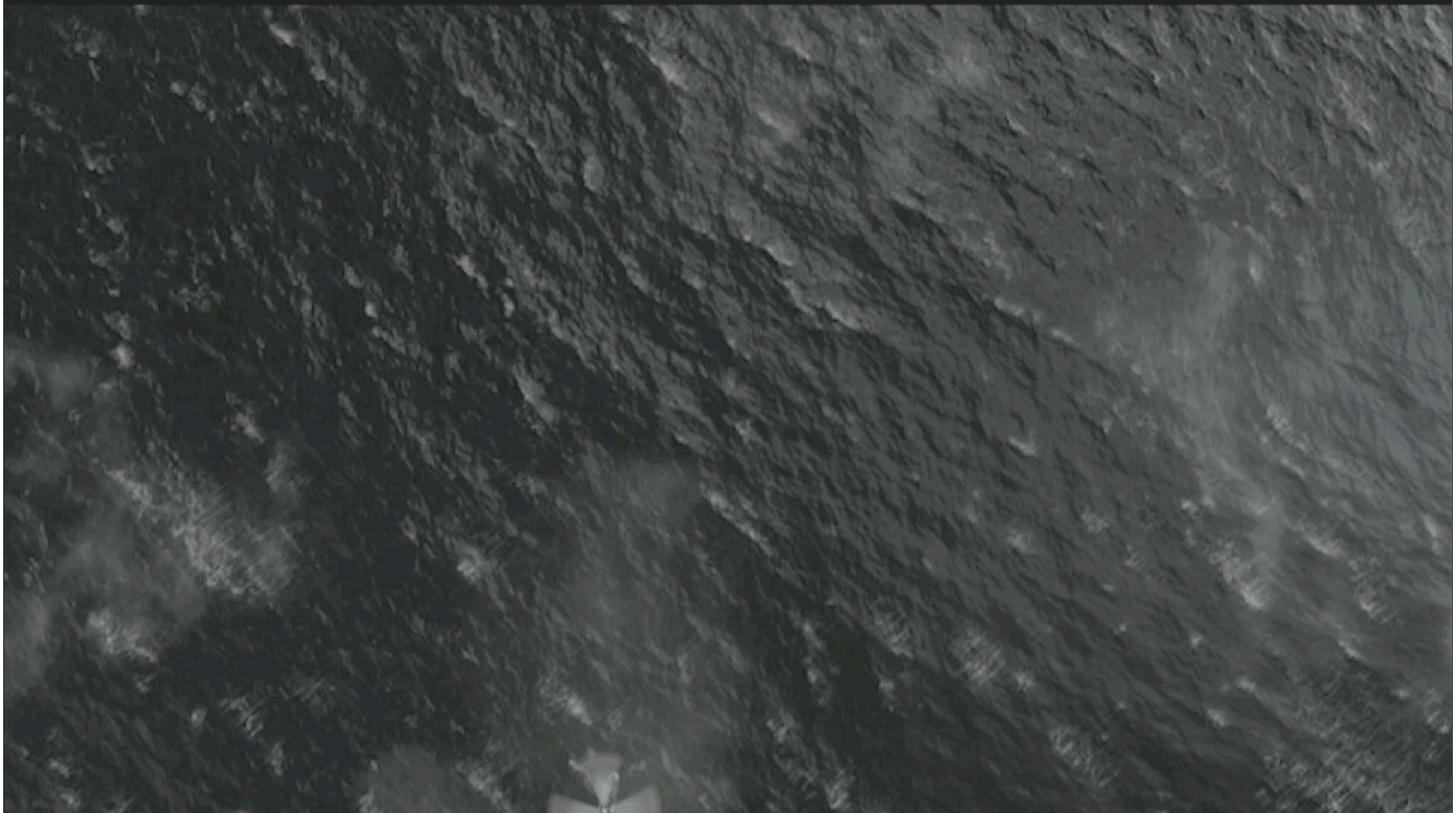


iWave Equation

- [Tessendorf 2004]

$$\frac{\partial^2 \phi}{\partial t^2} = \sqrt{-\nabla^2}\phi$$

Superman Returns (2006)



Surf’s Up (2007)

Happy Feet (2006)

1163 #653354 : user:slo sc49.08:CmpMain.Main-0046 - 15:03 Oct 02 iWave Equation:

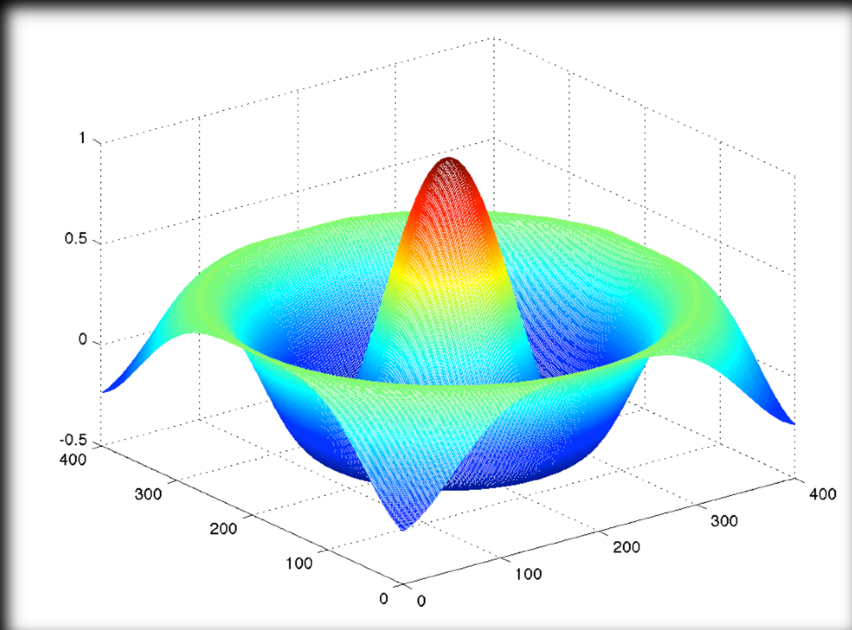
$$\frac{\partial^2 \phi}{\partial t^2} = \sqrt{-\nabla^2}\,\phi$$

Wave Equation:

$$\frac{\partial^2 \phi}{\partial t^2} = c\nabla^2 \phi$$

Fractional Laplacian

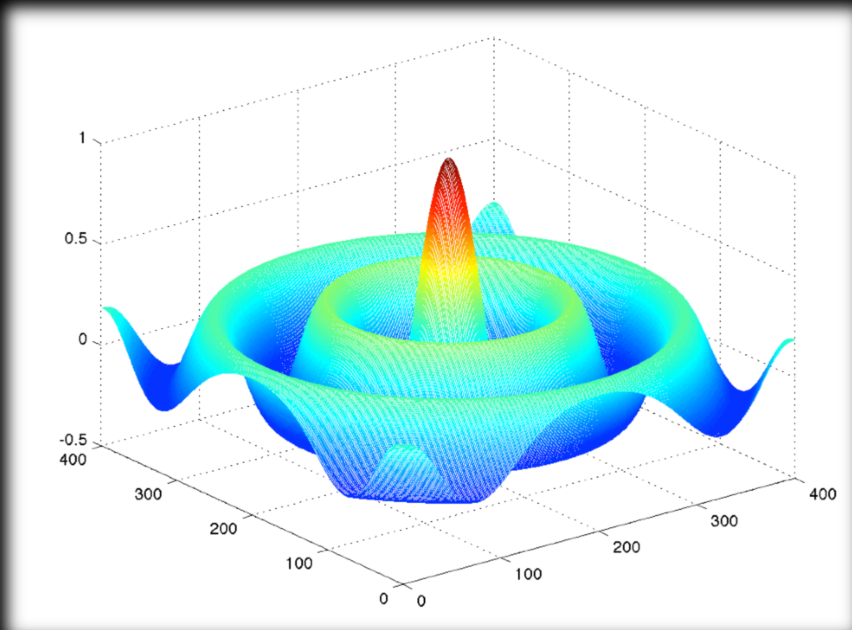
- Scaled sum of Bessel functions:

$1^2 e^{-\sigma^2}$  $+$

Fractional Laplacian

- Scaled sum of Bessel functions:

$$2^2 e^{-4\sigma^2}$$

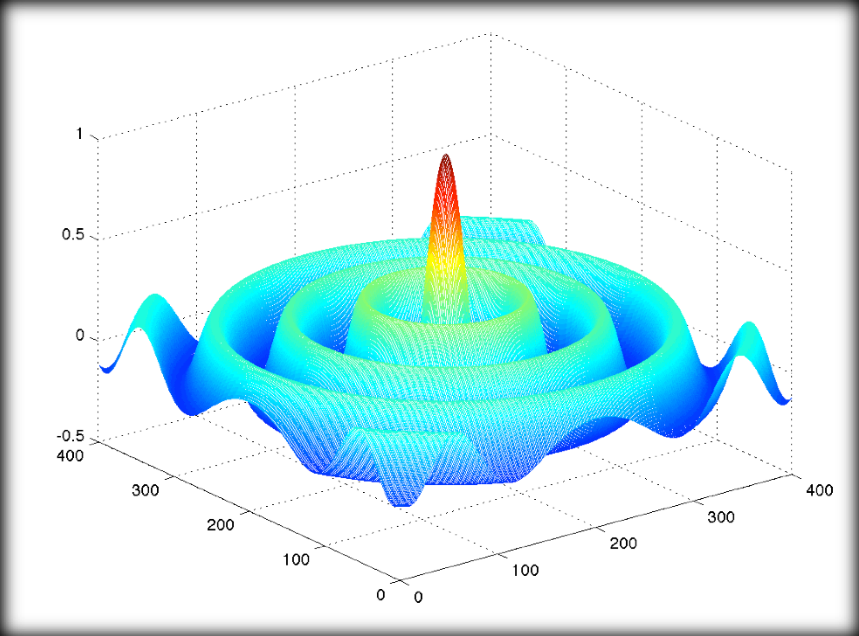


$+$

Fractional Laplacian

- Scaled sum of Bessel functions:

$$3^2 e^{-9\sigma^2}$$

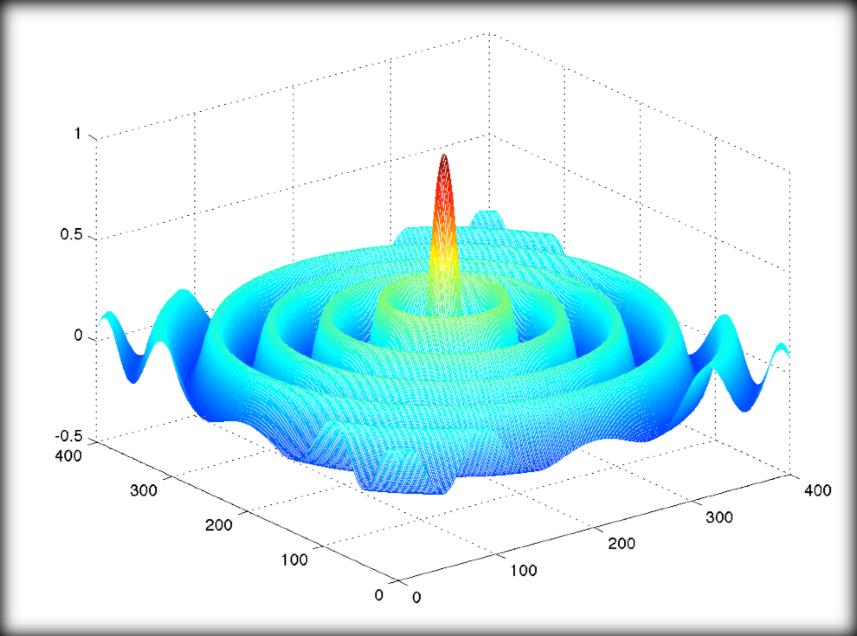


$+$

Fractional Laplacian

- Scaled sum of Bessel functions:

$4^2 e^{-16\sigma^2}$

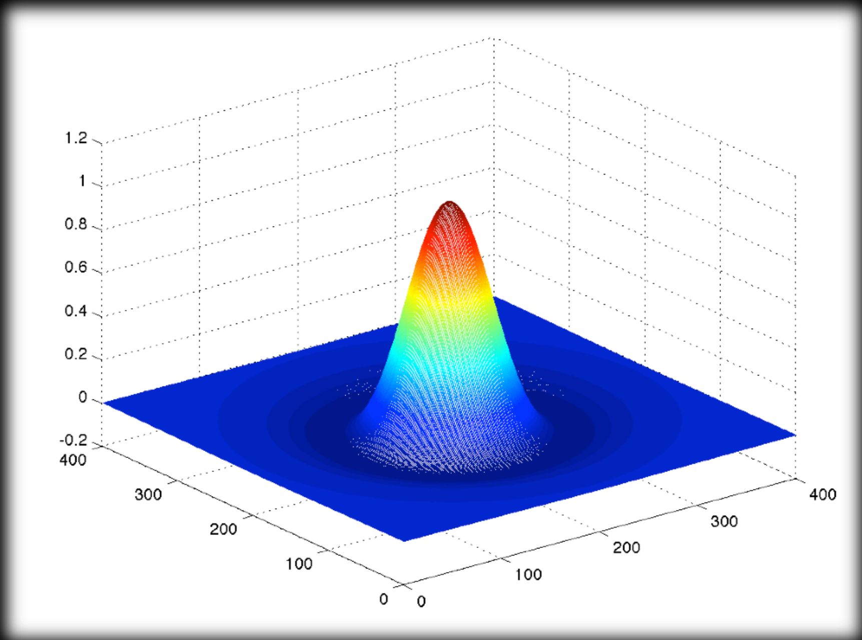


$+$

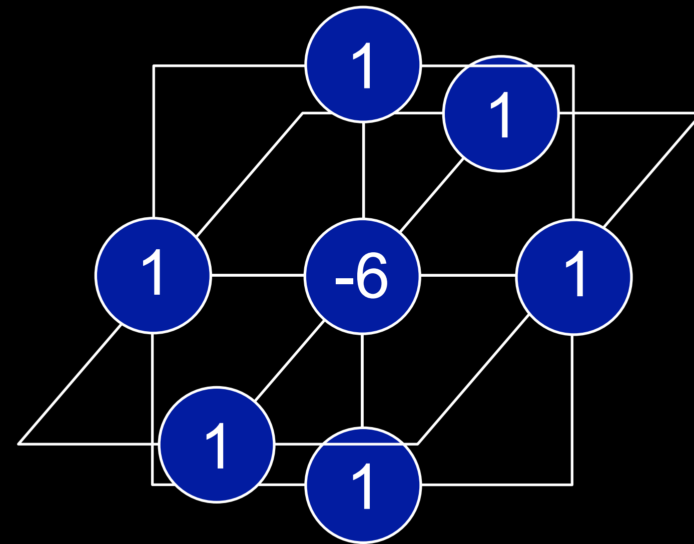
Fractional Laplacian

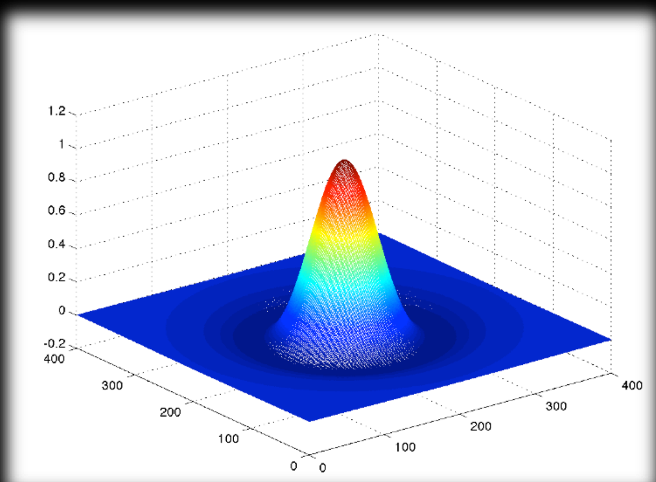
- Scaled sum of Bessel functions:

... =

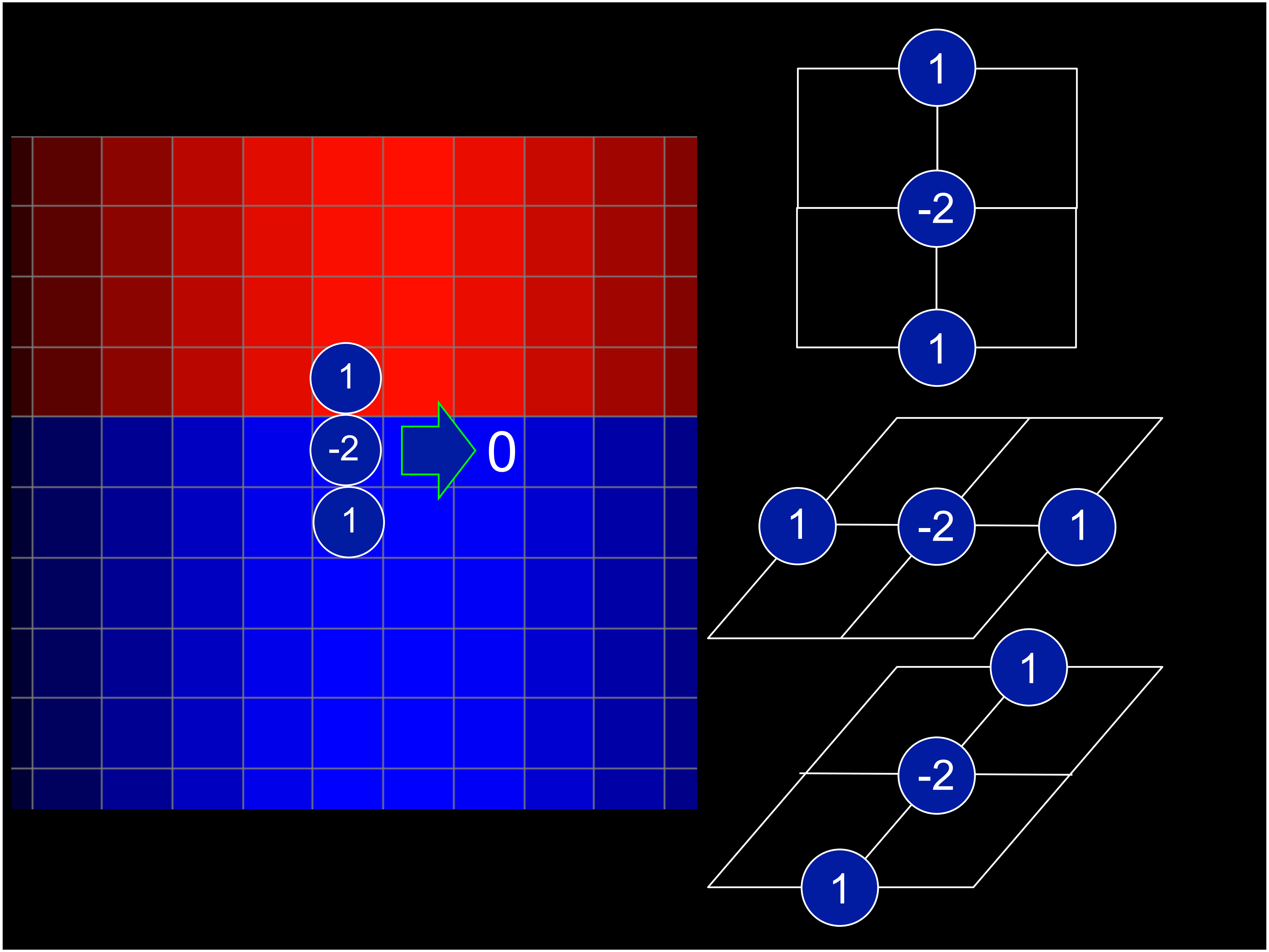


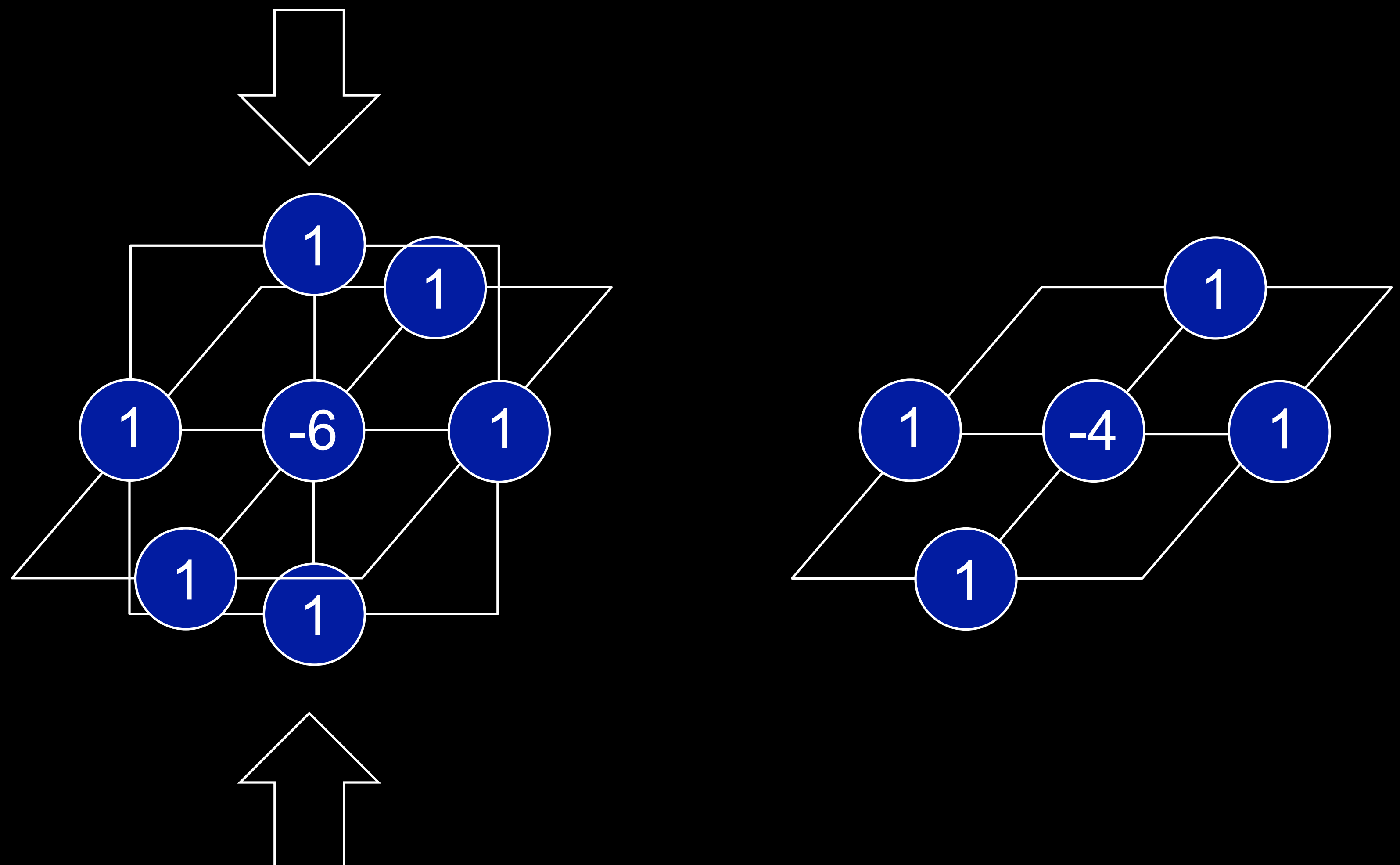
	1	
1	-4	1
	1	



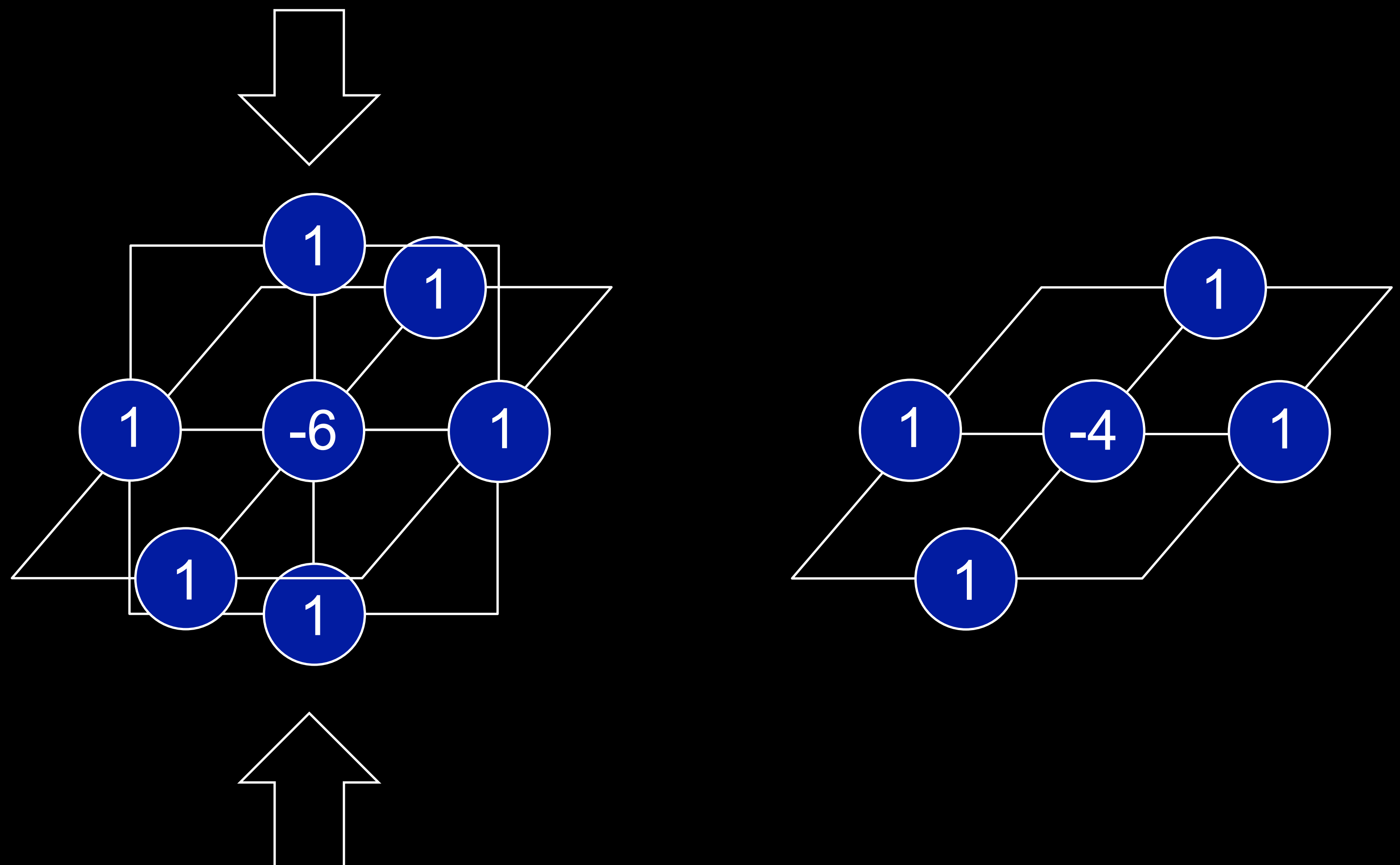


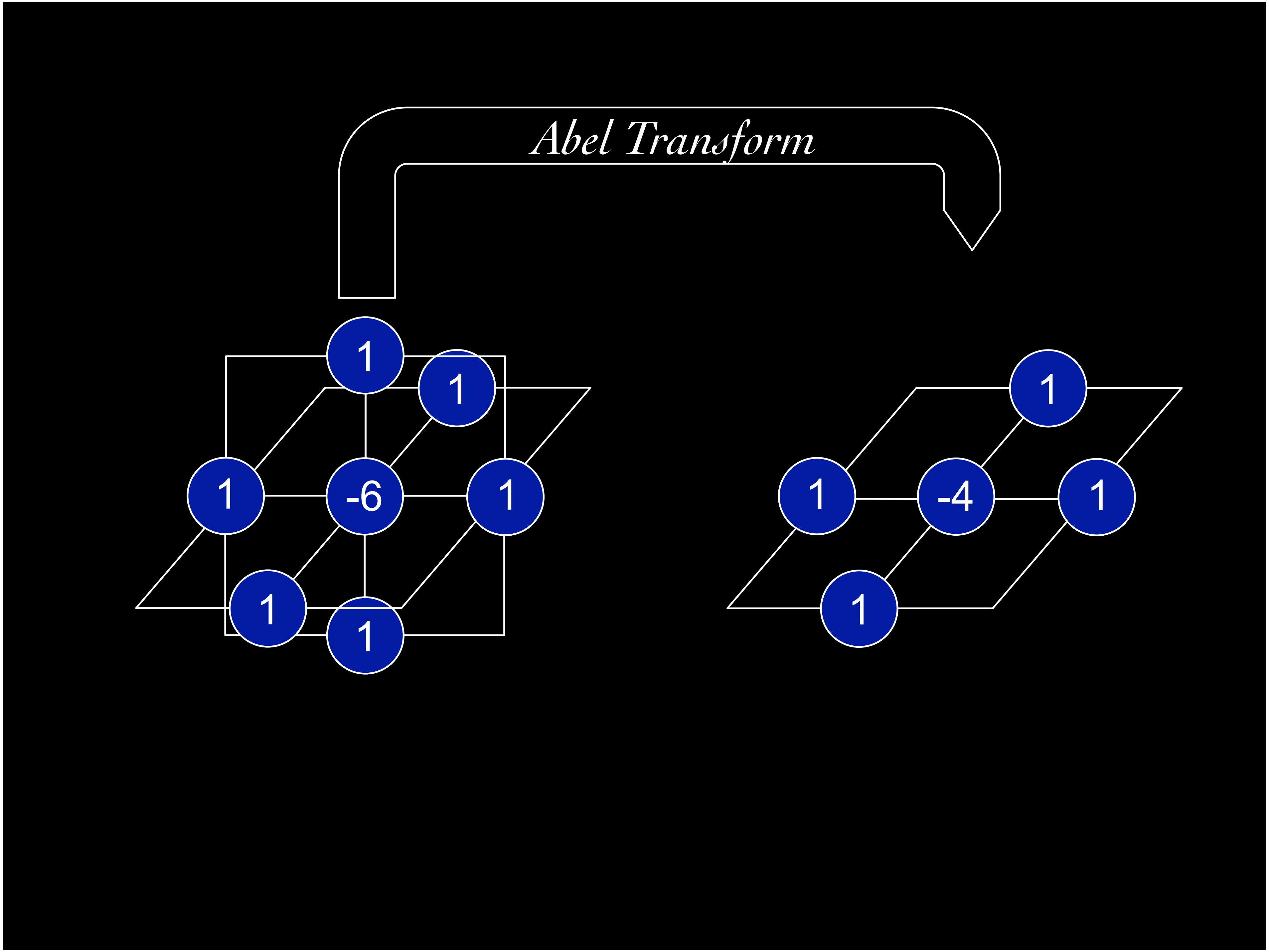
???


1
-2
1
0
1
-2
1
1
-2
1
1
-2
1

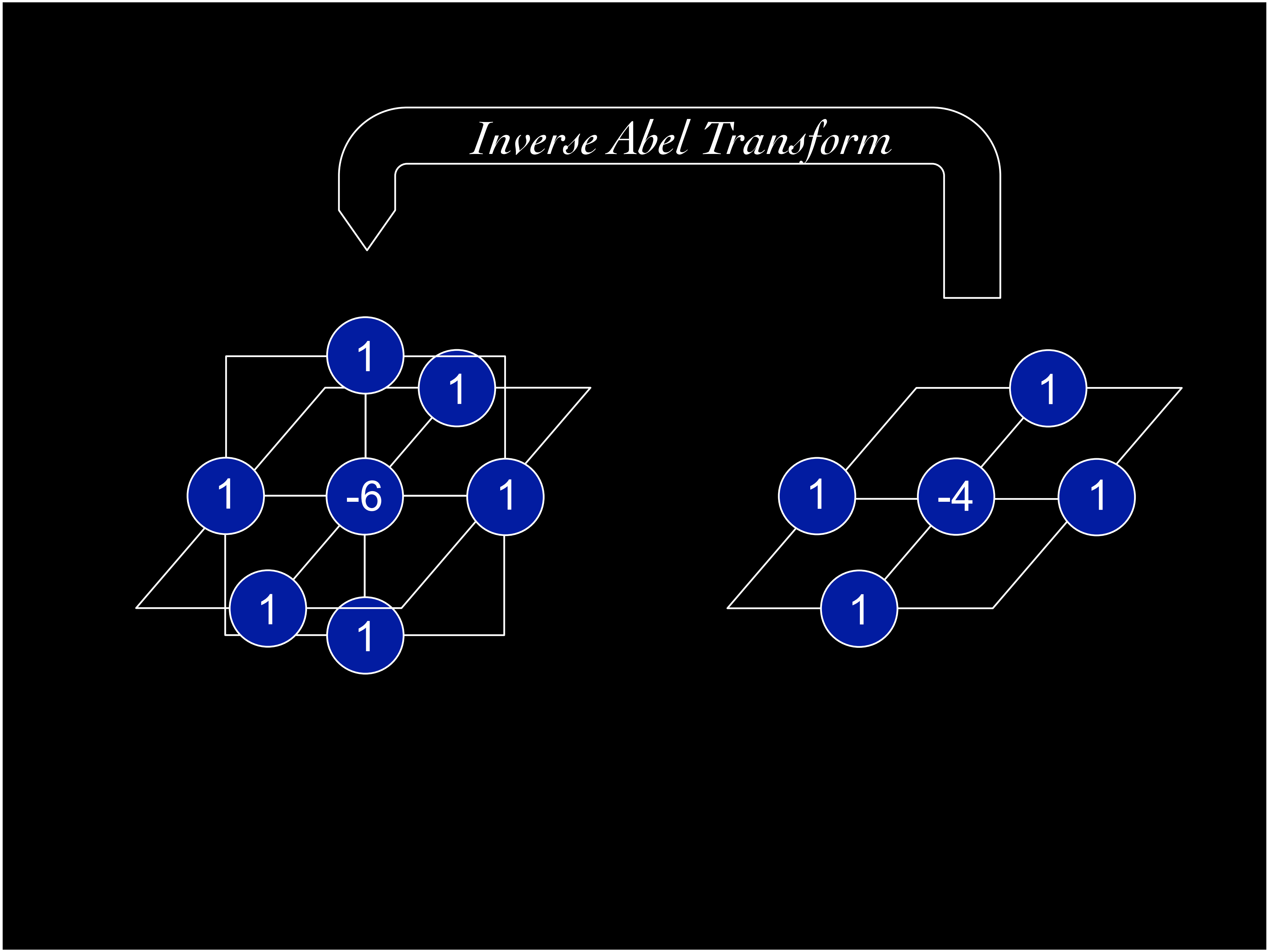

1
1
1
-6
1
1
1
1
1
-4
1
1

This is an *Abel* transform





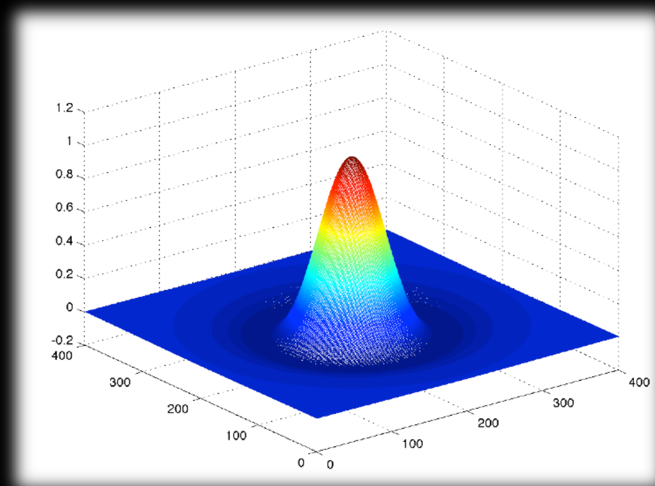
Abel Transform
1
1
1
-6
1
1
1
1
1
-4
1
1



Inverse Abel Transform
1
1
1
-6
1
1
1
1
-4
1
1
1

Inverse Abel Transform

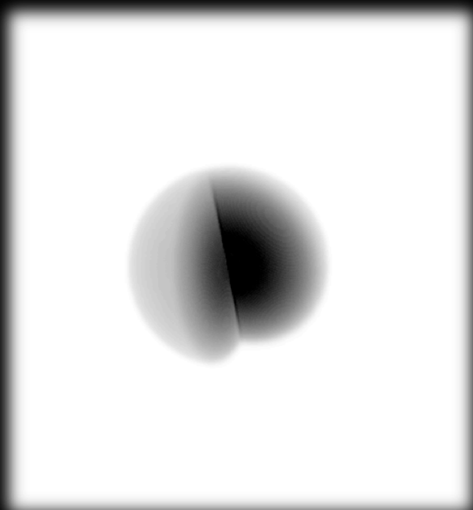
???

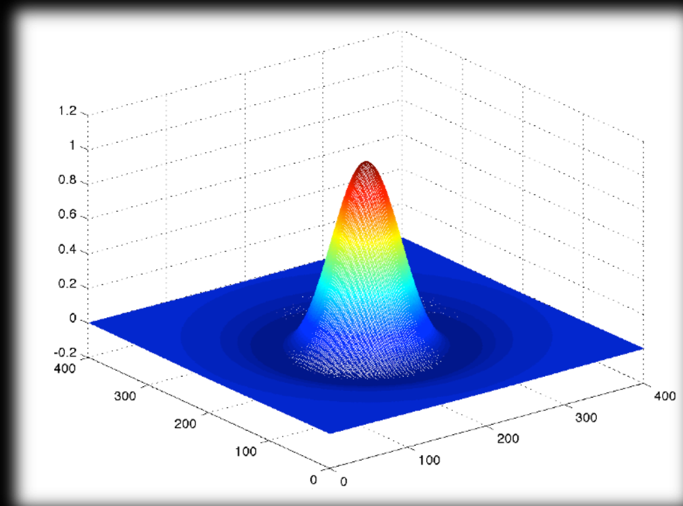


Inverse Abel Transform

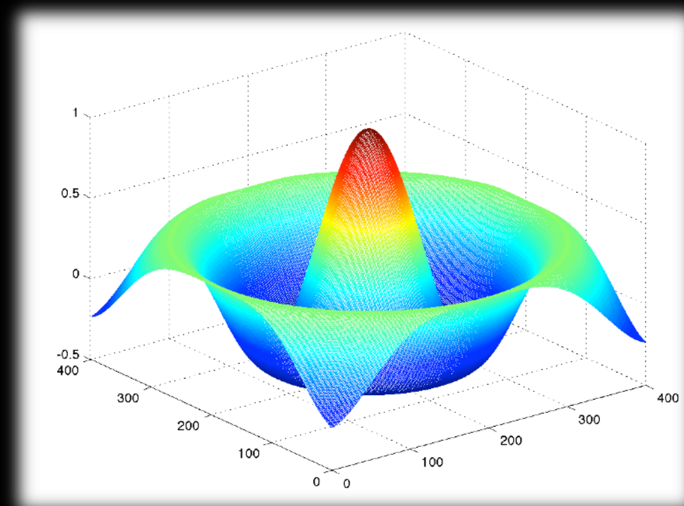
$$\frac{1}{\pi}\int k^3 e^{-k^2}\,\mathrm{sinc}(kr)\,dk \qquad \int k^2 e^{-k^2} J_0(kr)\,dk$$

Inverse Abel Transform

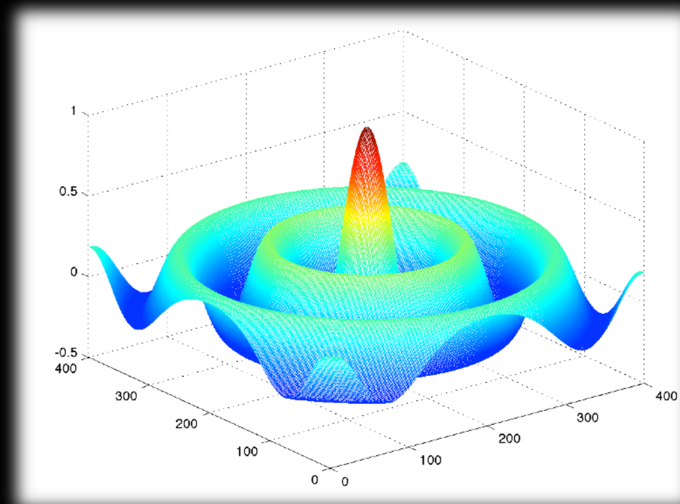




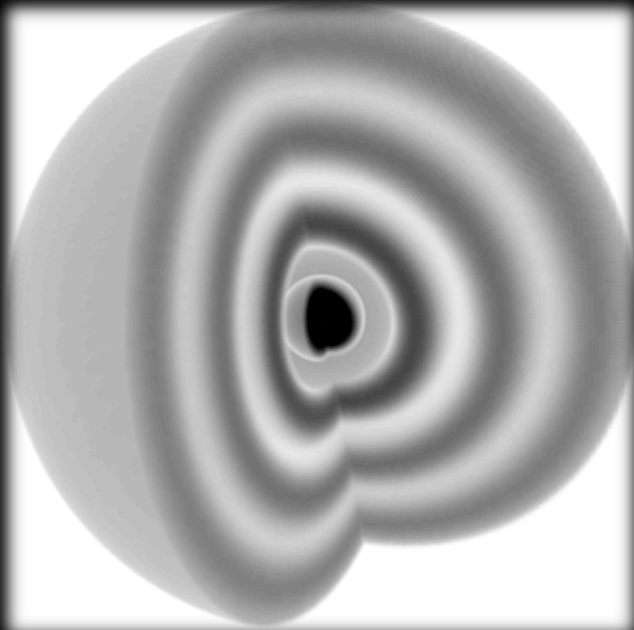
Inverse Abel Transform

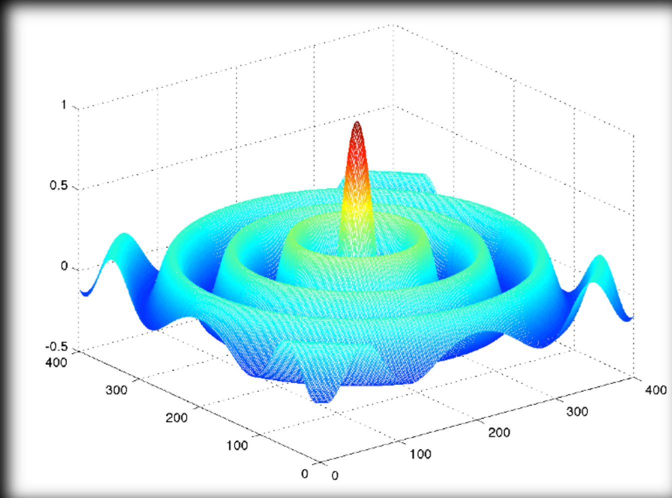


Inverse Abel Transform

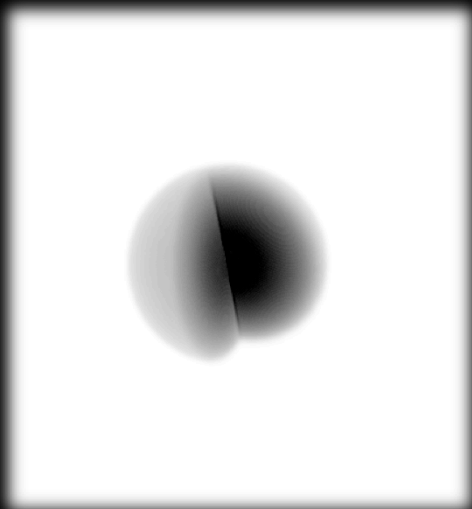


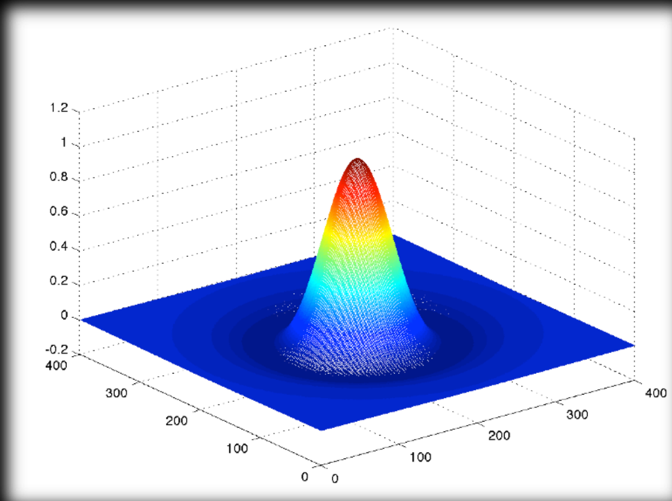
Inverse Abel Transform





Inverse Abel Transform





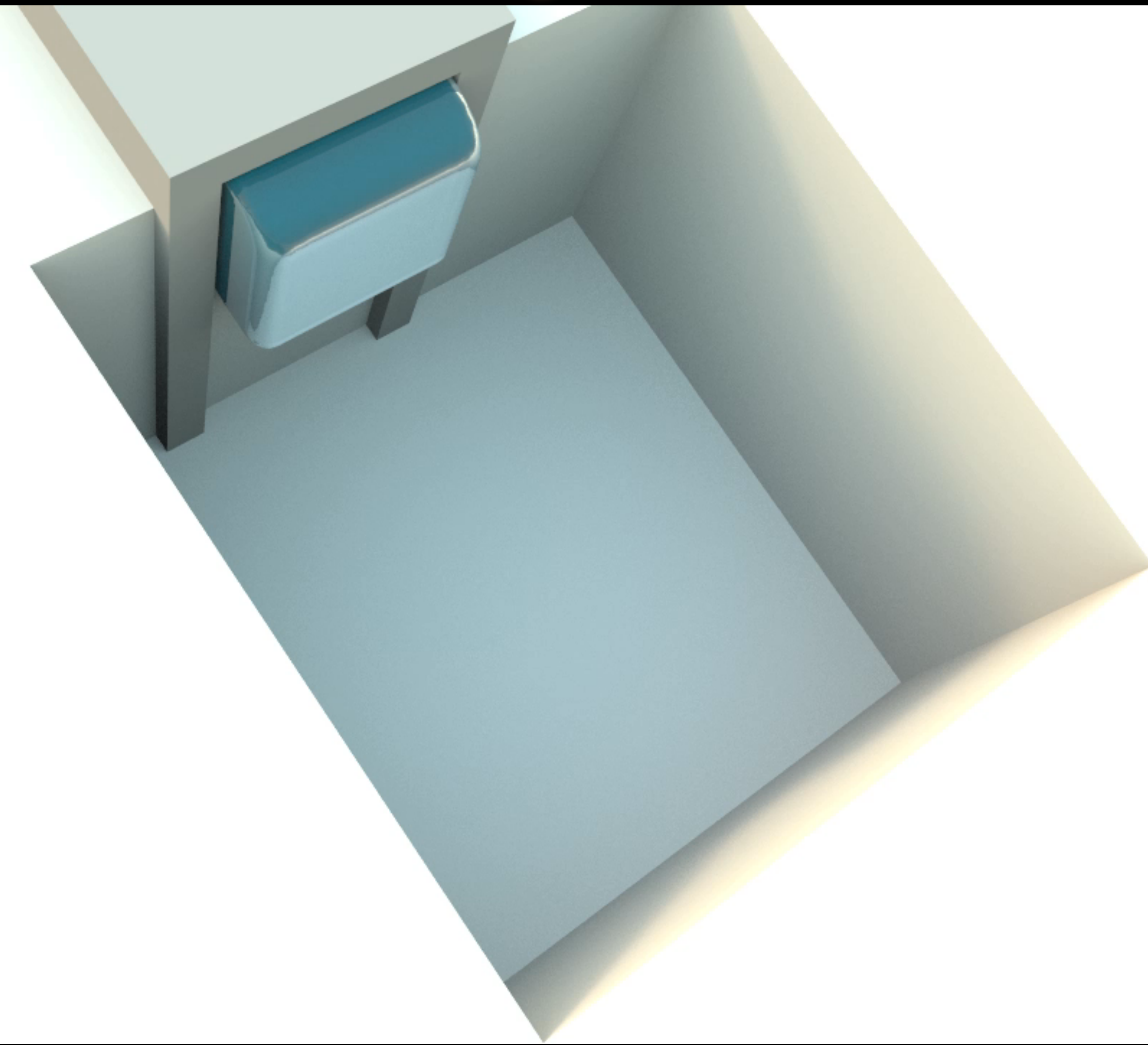
Finding

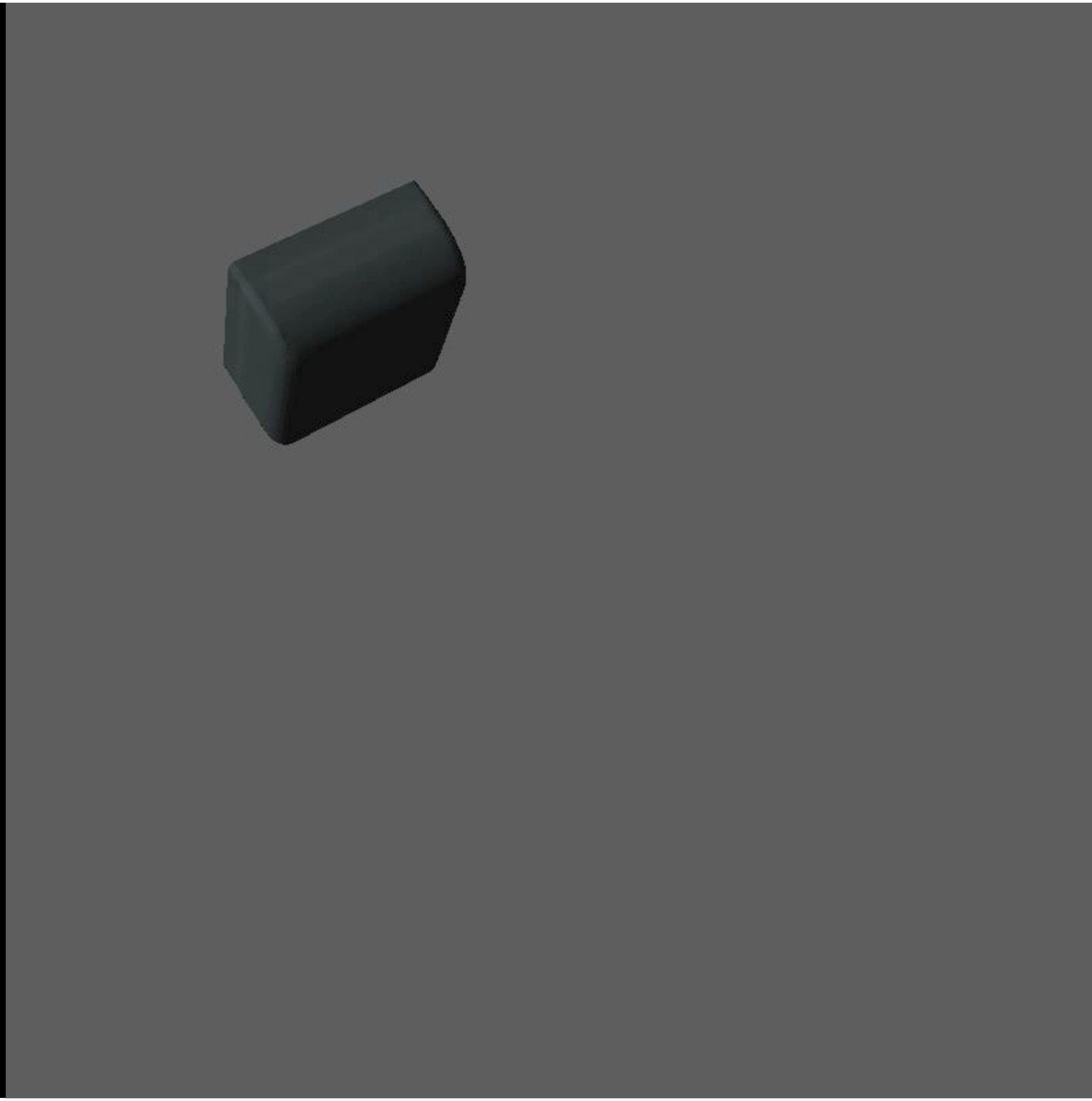
- *Any* 2D operator can be applied to the Closest Point Method by taking its *inverse Abel transform*

Outline

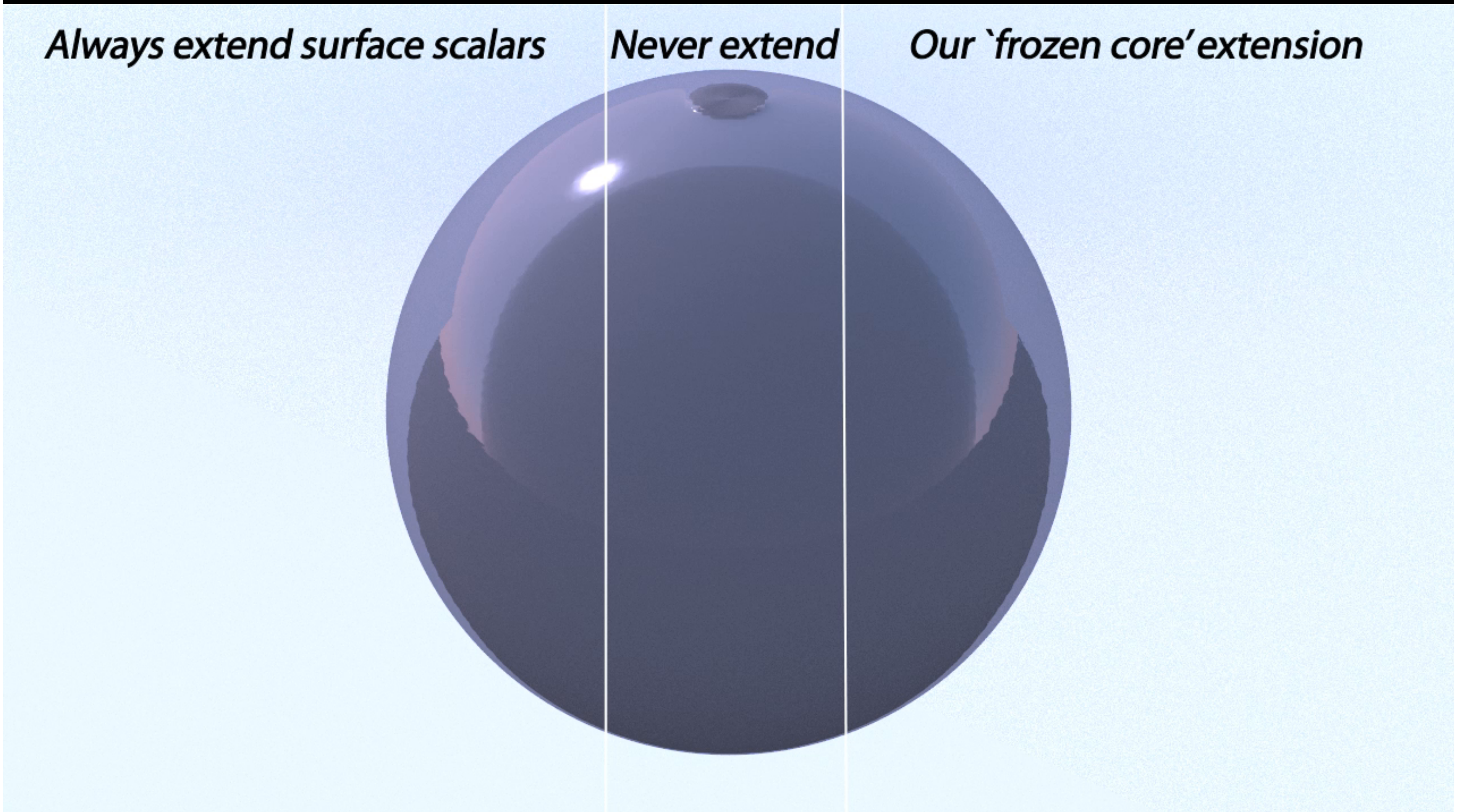
- Previous Works
- The Closest Point Method
- 3D iWave
- Additional Extensions
- Results

Turbulence Seeding





Extension Field



Advection

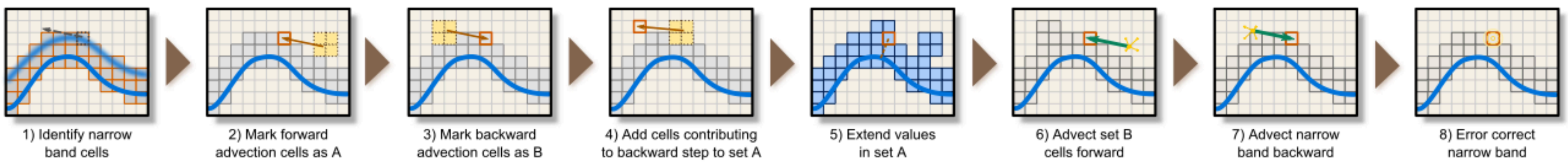
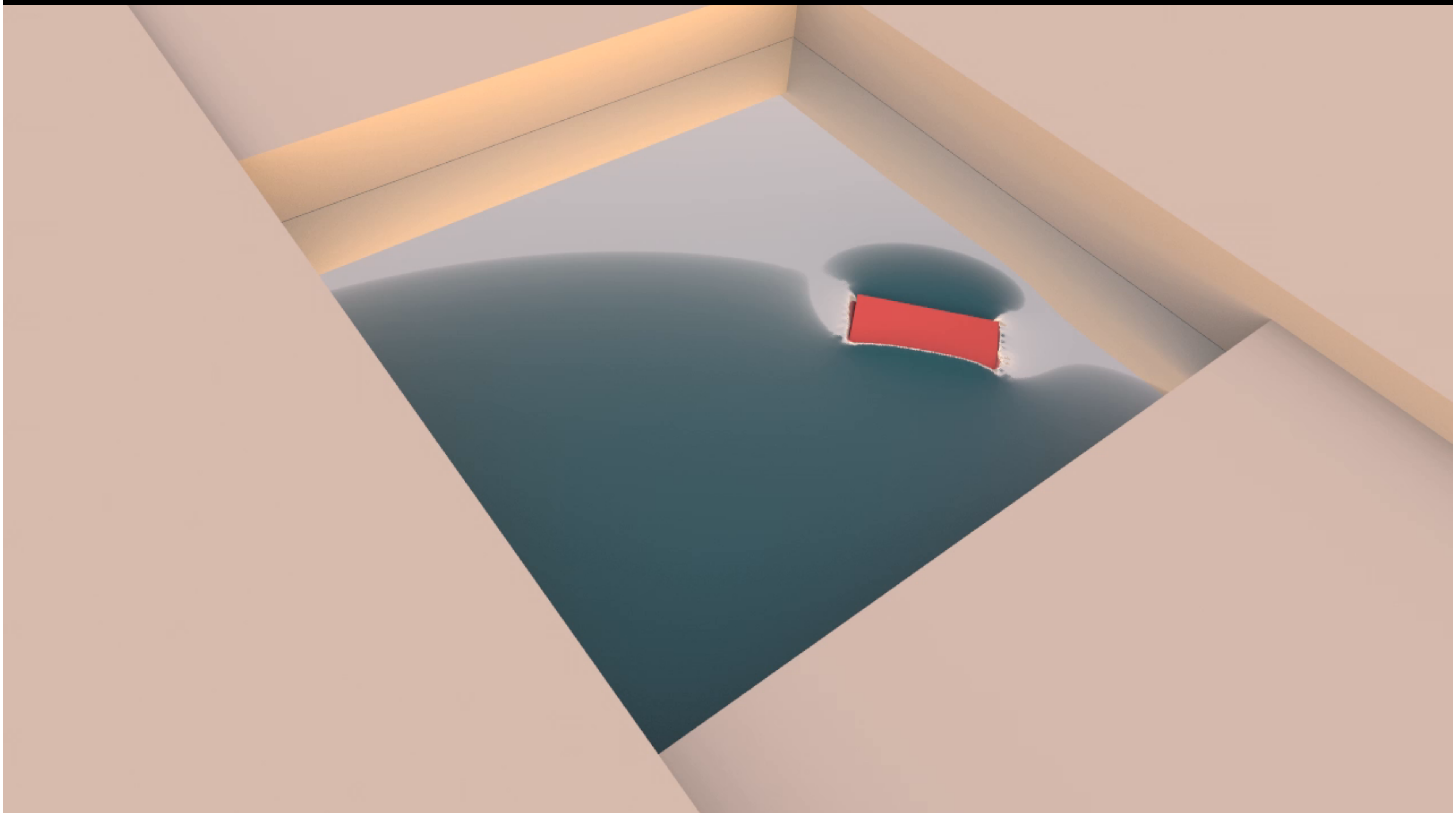


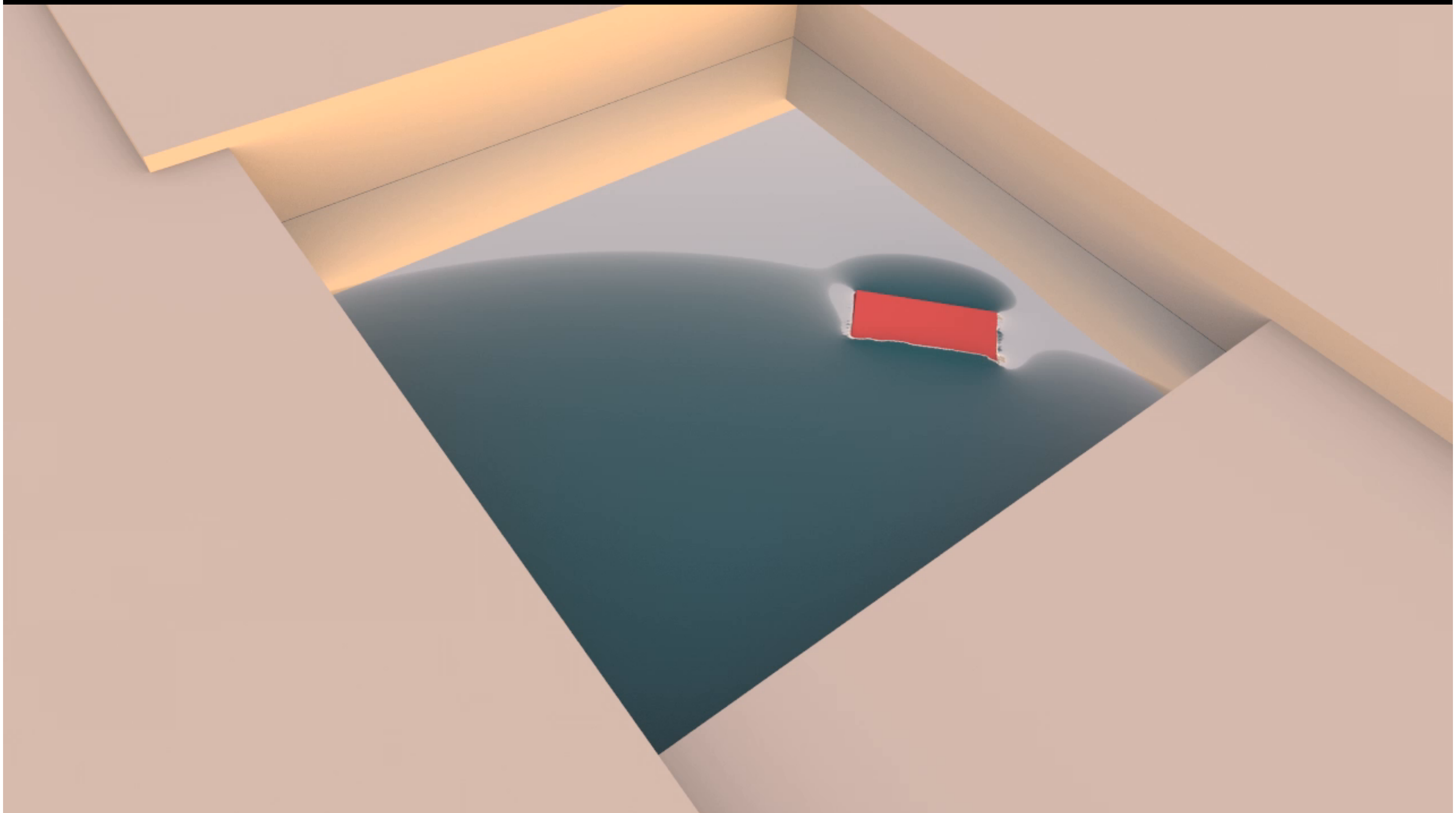
Fig. 5: An overview of our narrow band advection. For clarity, we only show one side of the narrow band, and use linear interpolation stencils. The active cells for each step are highlighted, and the source cell with its velocity is shown in step 1. Note that both step 2 and 4 are adding cells to set *A*, which contains all cells that are initialized by extension in step 5. The cells from step 4 are needed to compute the backward advection in step 7.

Outline

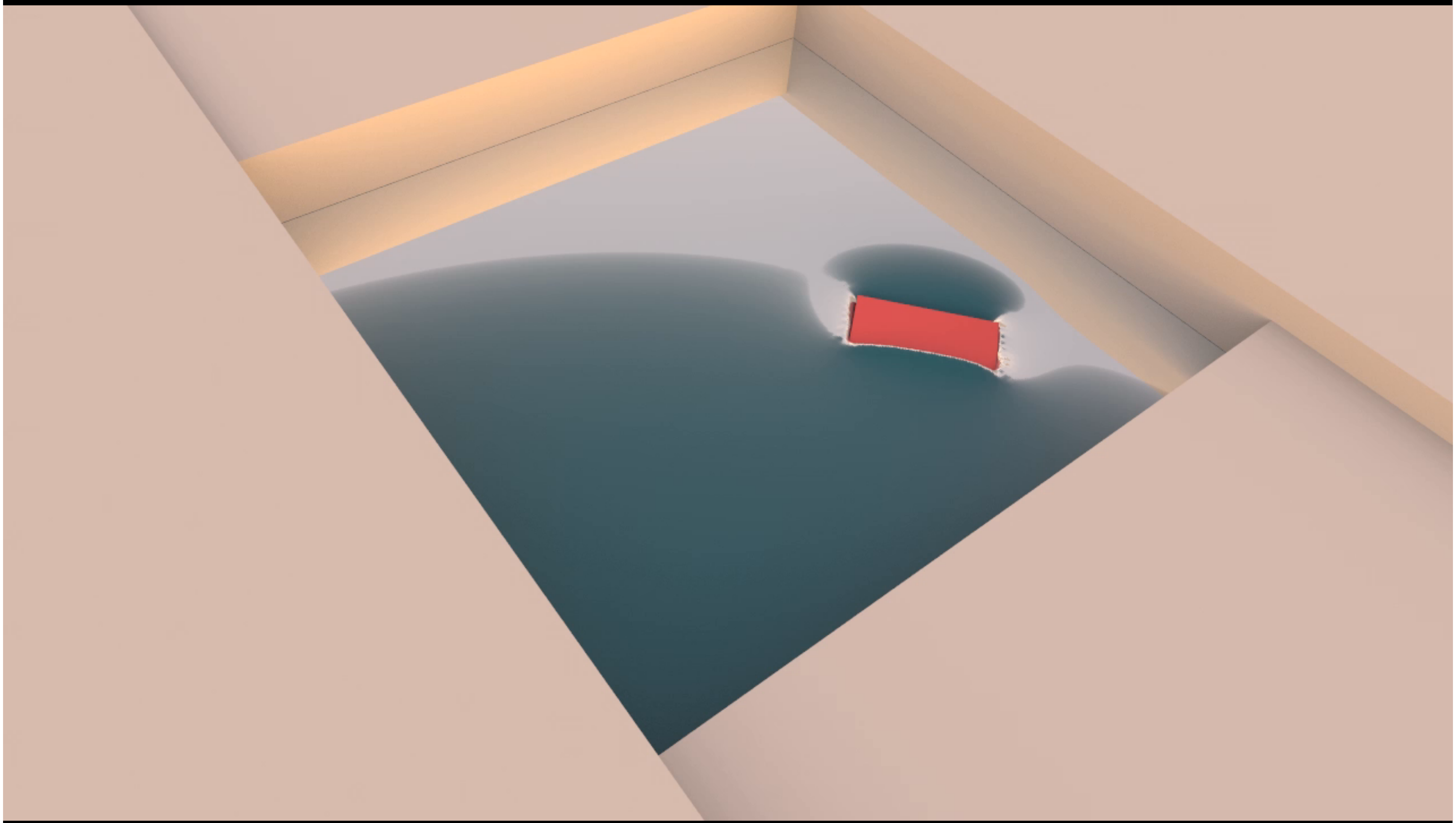
- Previous Works
- The Closest Point Method
- 3D iWave
- Additional Extensions
- Results



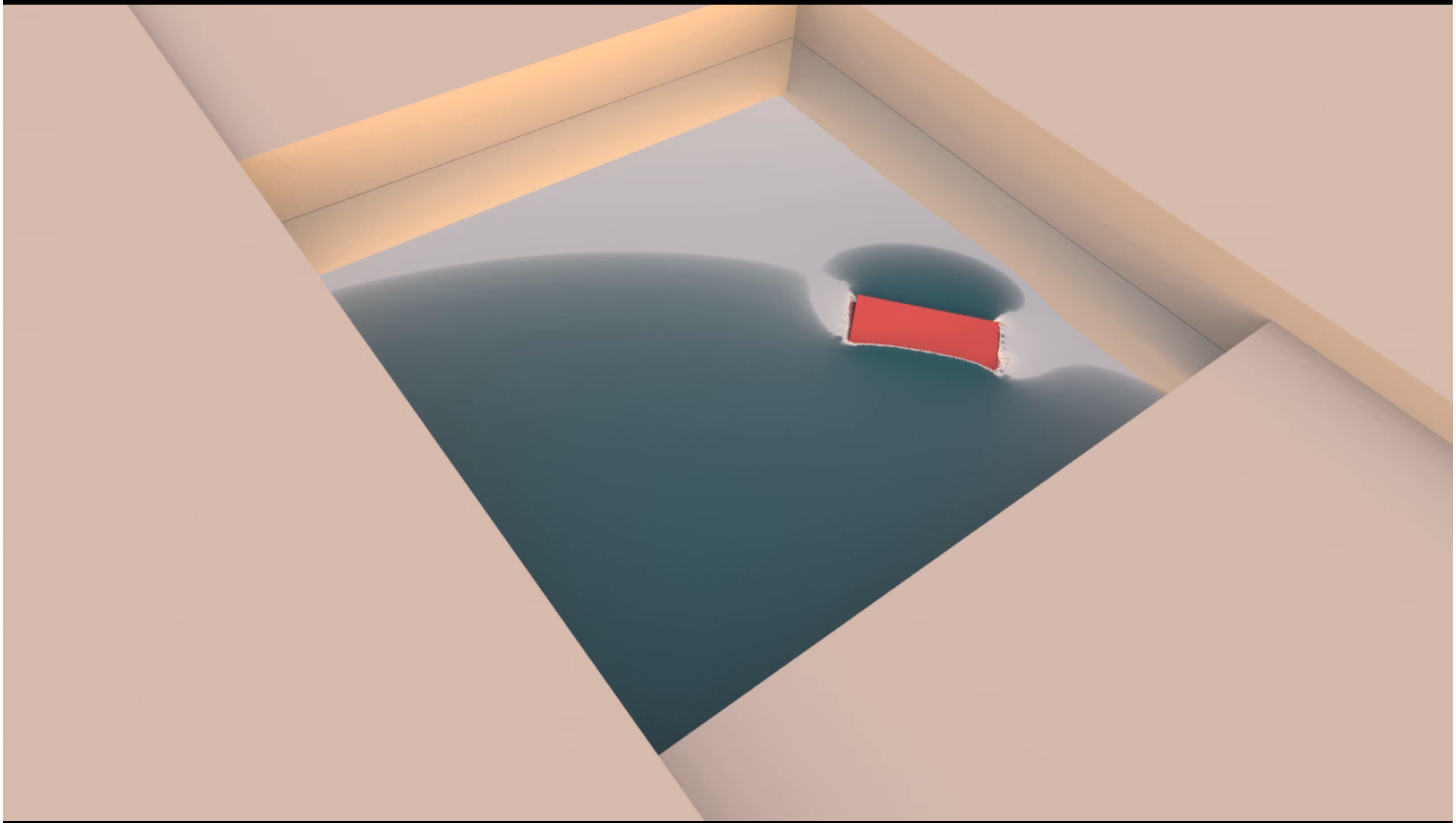
100^3 Houdini simulation, 00:12 per frame



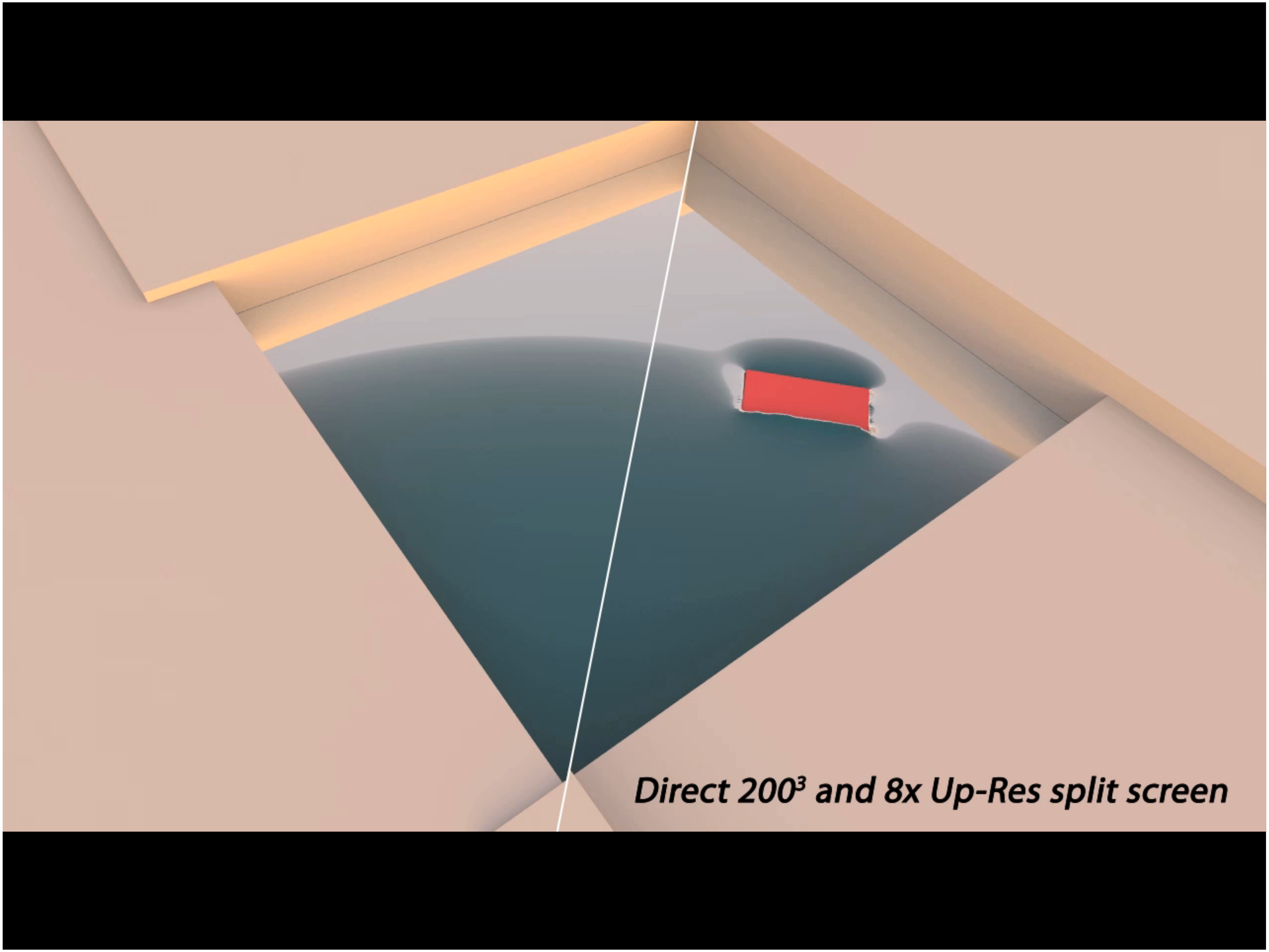
200^3 Houdini simulation, 03:44 per frame



4x up-res, 00:58 per frame



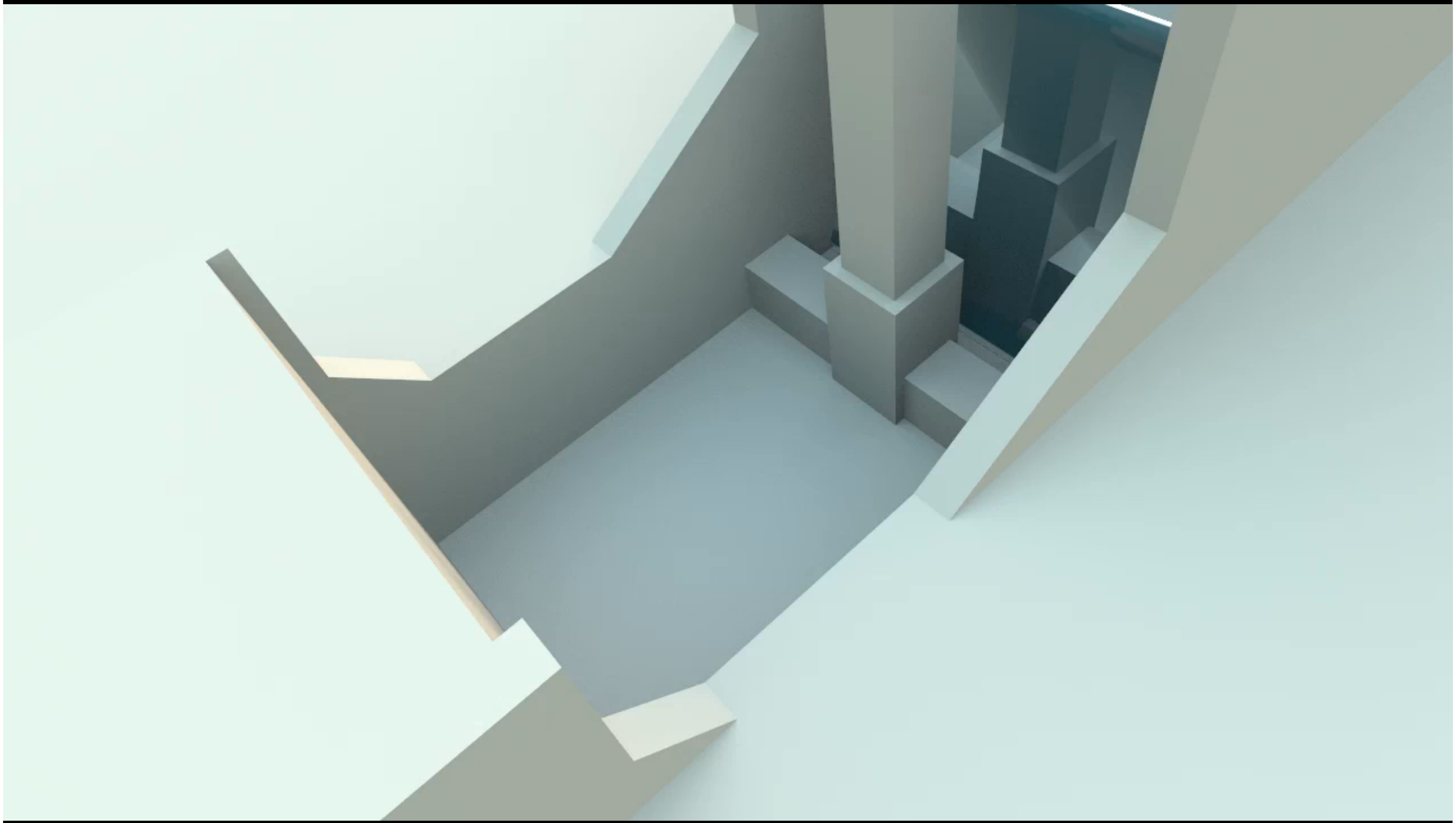
8x up-res, 05:29 per frame


Direct 200^3 and 8x Up-Res split screen

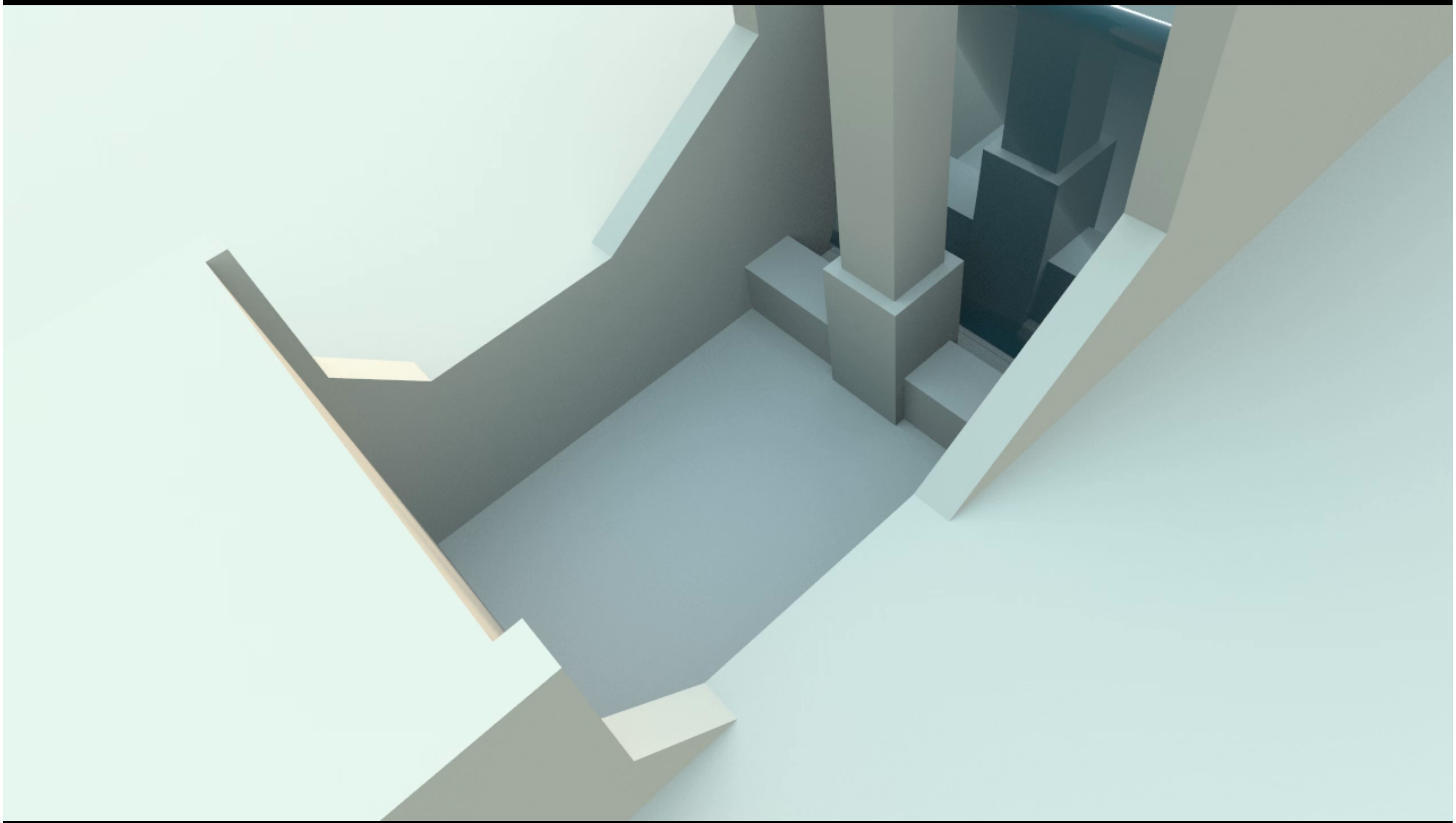
Performance Per Frame

	Houdini, Direct	Our Algorithm
100^3	00:24	N/A
200^3	03:44	00:12
400^3	38:00	00:58
800^3	N/A	05:29
1000^3	N/A	11:12

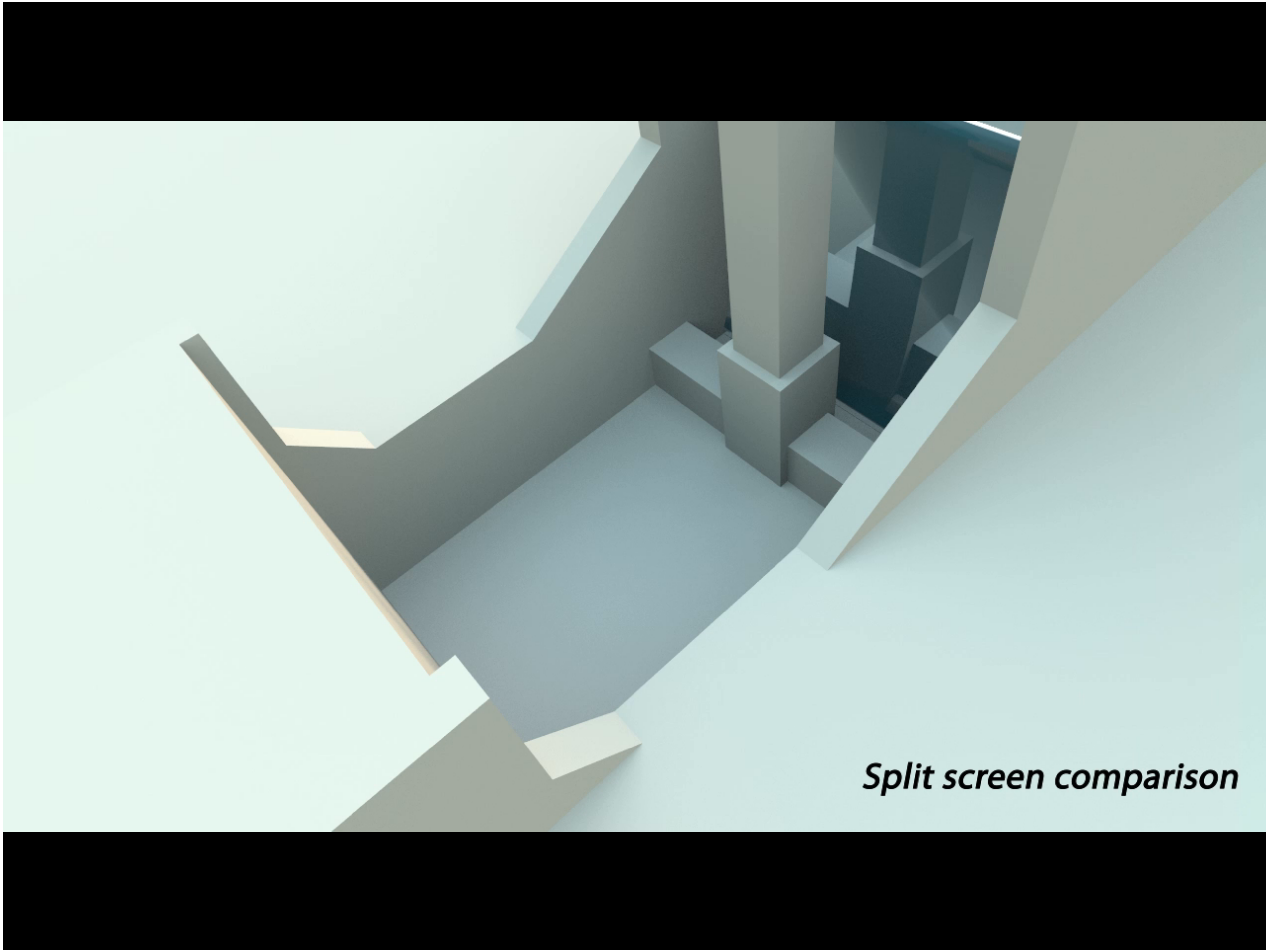
12-core, 2.66 Ghz Intel Xeon

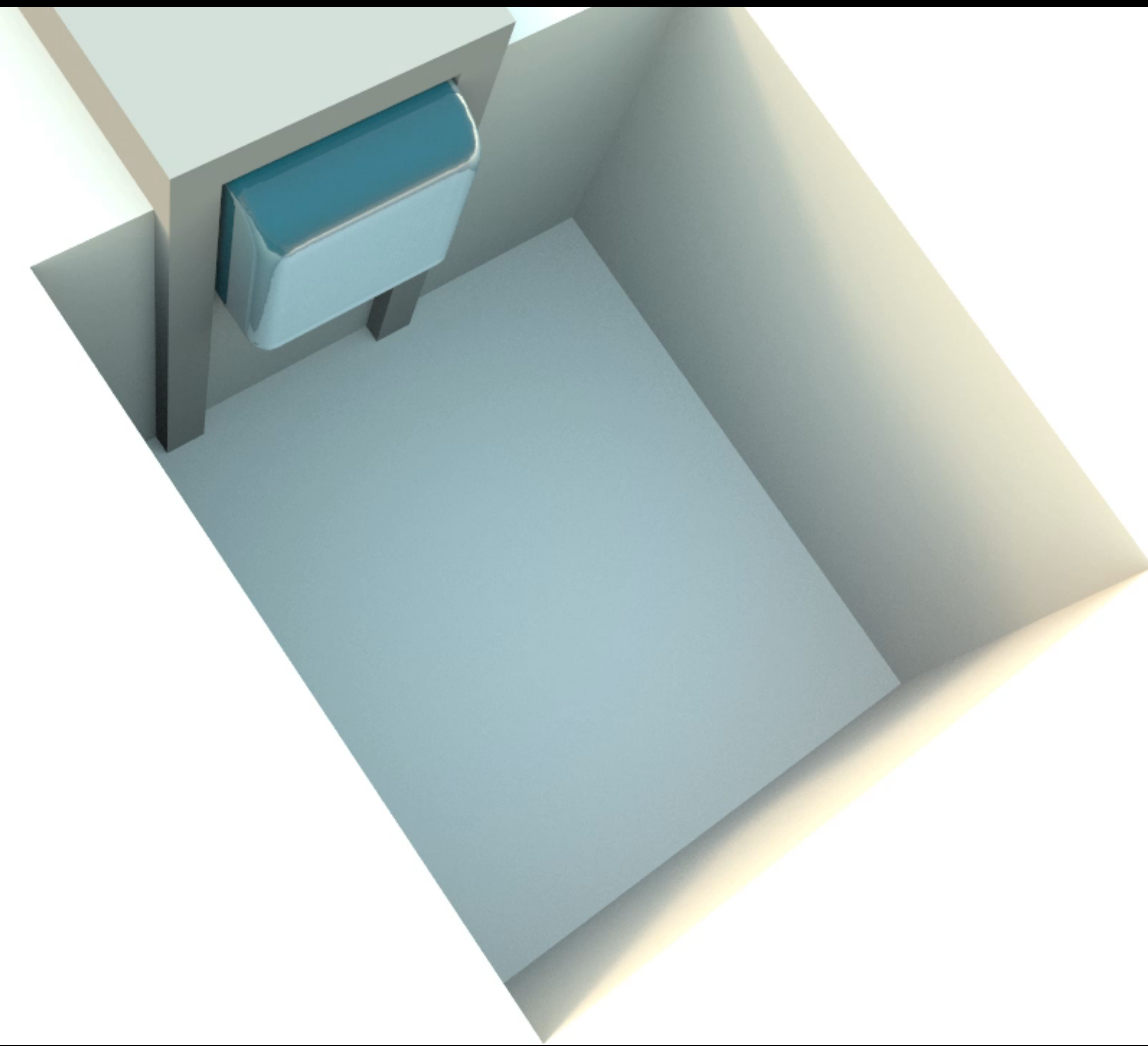


100^2 x 50 simulation, 00:18 per frame

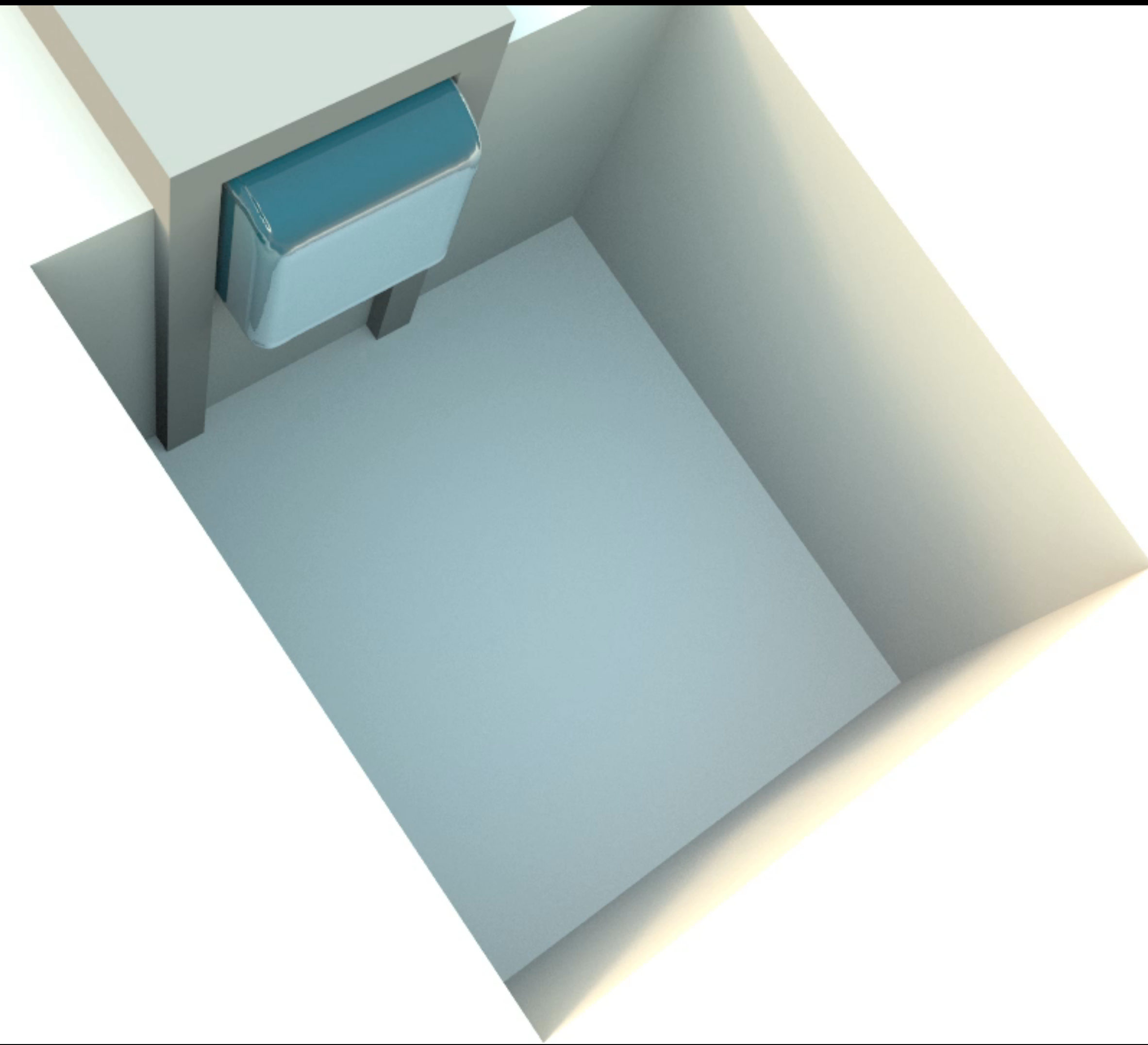


8x up-res, 02:48 per frame

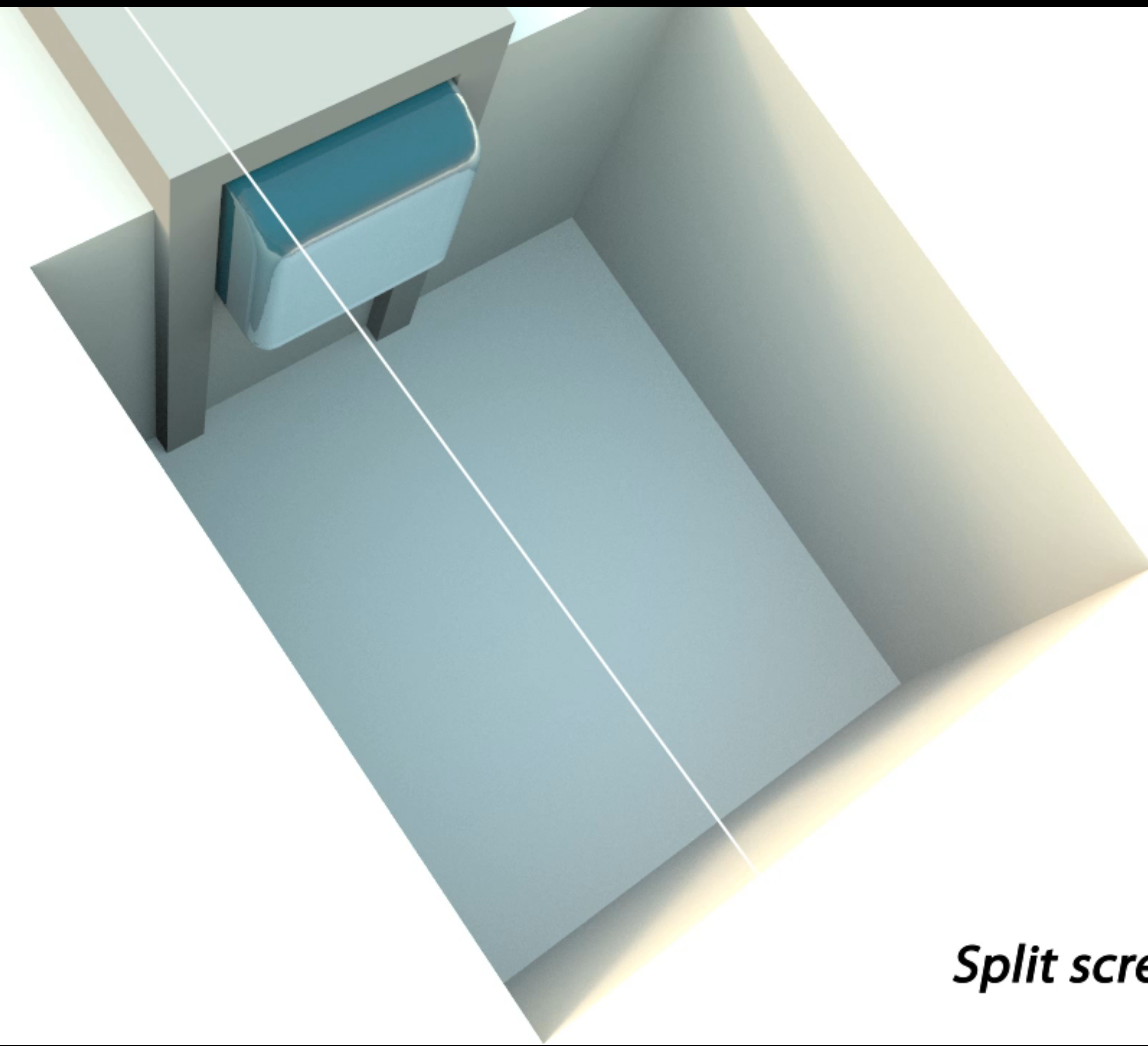

Split screen comparison



100^3 simulation, 01:52 per frame



8x upres, 03:15 per frame



Split screen comparison

Performance Per Frame

	Dam Break	Pouring
Original Simulation	00:18	01:52
2x Upres	00:06	00:07
4x Upres	00:21	00:34
8x Upres	02:48	03:15

12-core, 2.66 Ghz Intel Xeon

Performance Per Frame

	Dam Break	Pouring
Original Simulation	00:18	01:52
2x Upres	00:06	00:07
4x Upres	00:21	00:34
8x Upres	02:48	03:15

12-core, 2.66 Ghz Intel Xeon

Conclusions

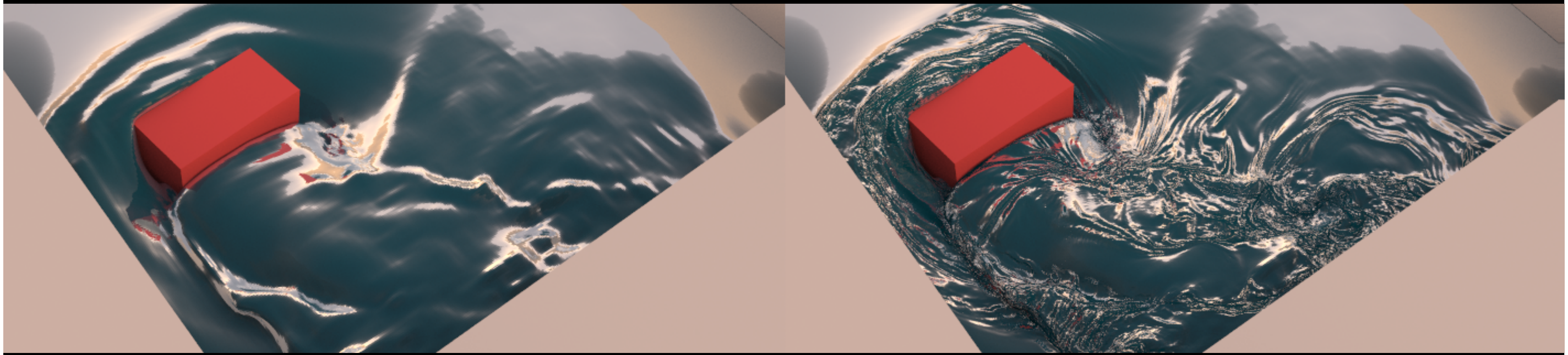
- Up-res liquids with a *surface simulation*
- The Closest Point Method (CPM) is simple, robust Eulerian method
- Adapt 2D stencils for CPM using the *inverse Abel transform*
- More details in paper
  - Turbulence seeding
  - Frozen core extension

Future Work

- Implicit CPM integrator?
- Up-res FLIP liquids?
- What other operators can we apply?

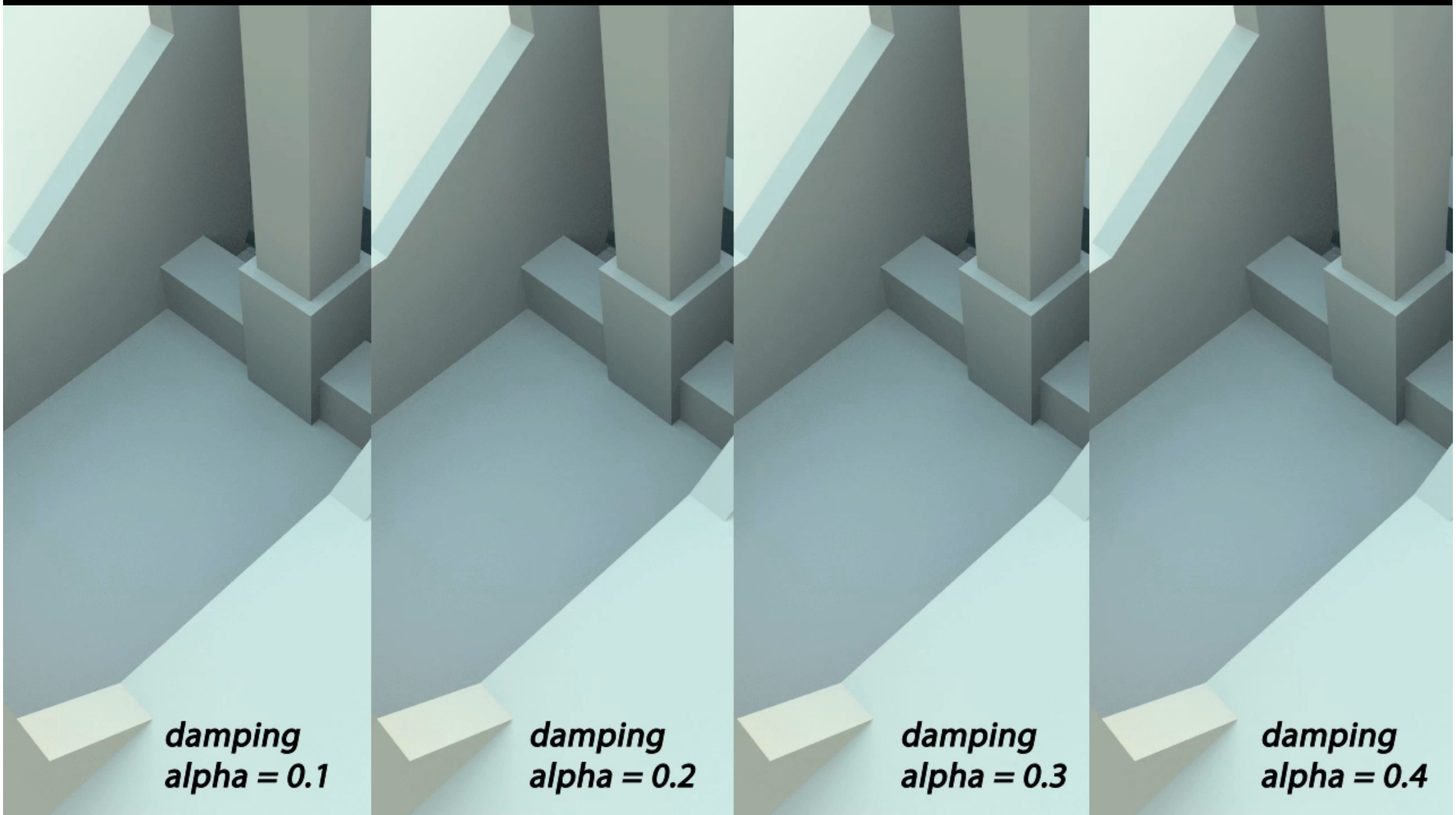
Acknowledgements

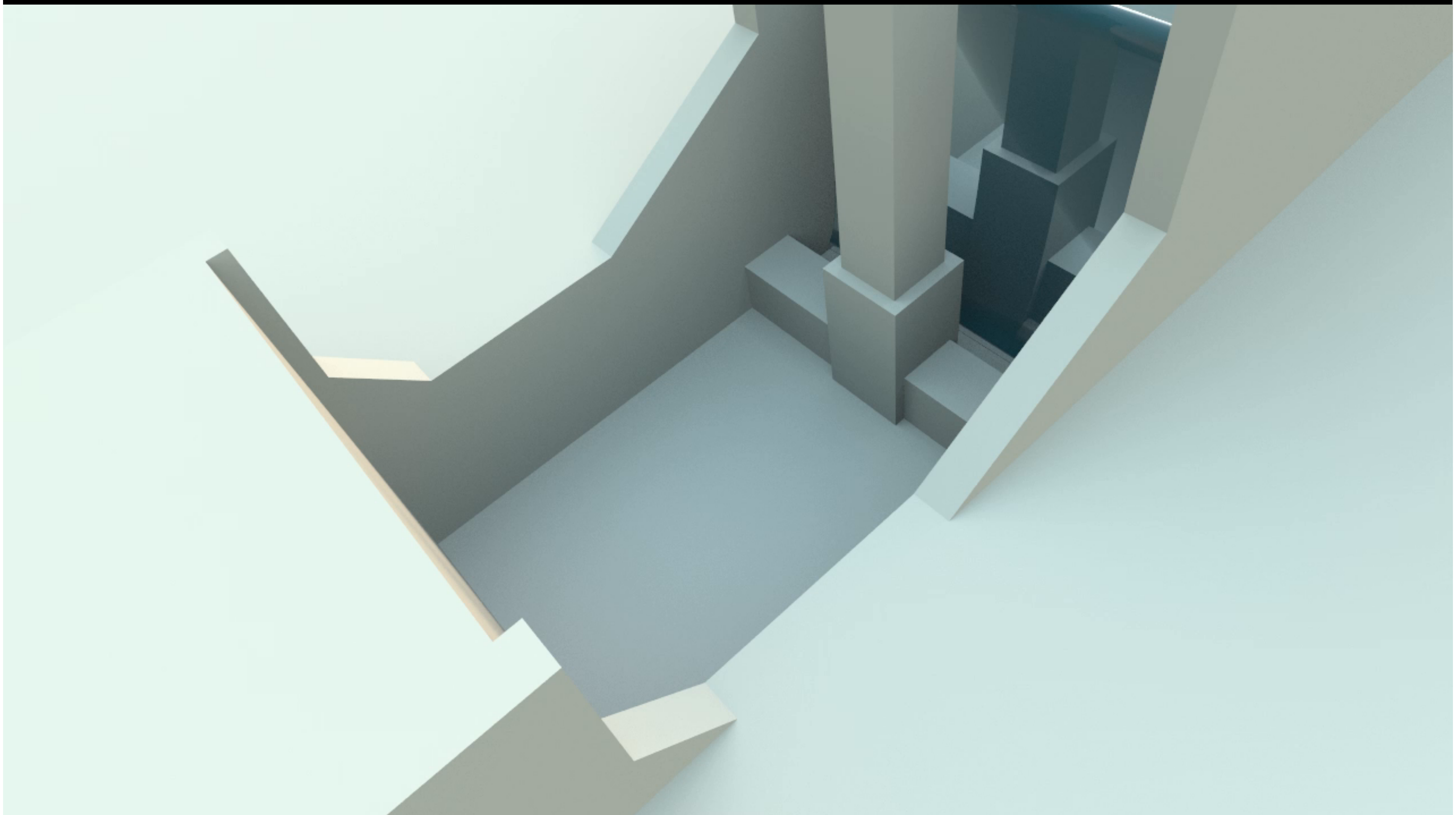
- NSERC Discovery, “Many-Core Physically Based Simulation”
- NSF CAREER, “Enabling Efficient Non-Linearities in Biomechanical Simulations”
- Side Effects Software, especially Jeff Lait



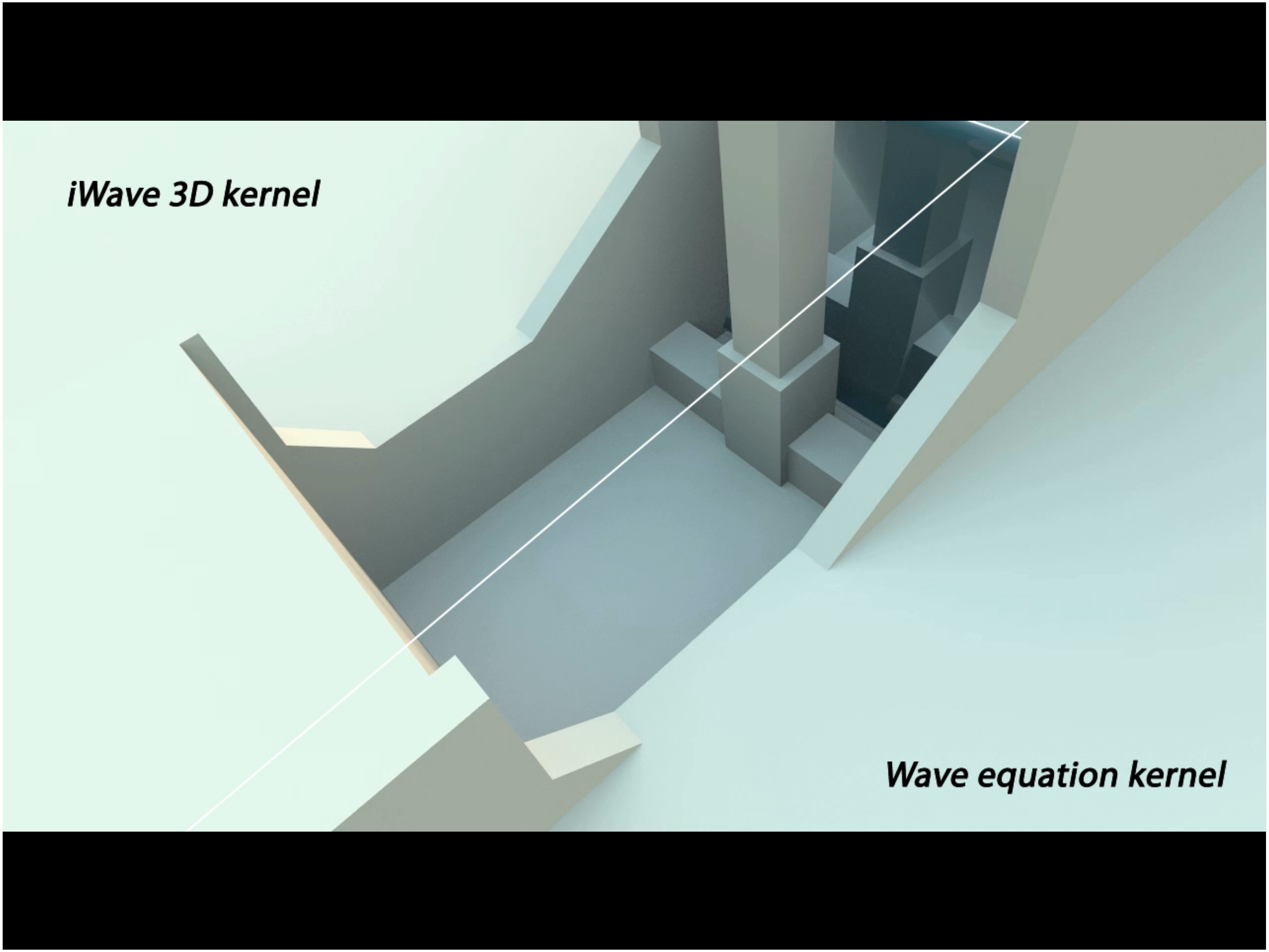
Thank You

Source: http://www.mat.ucsb.edu/~kim/CPT/source.html


damping
alpha = 0.1
damping
alpha = 0.2
damping
alpha = 0.3
damping
alpha = 0.4



8x up-res, classic wave equation


iWave 3D kernel
Wave equation kernel

Performance Per Frame

	Houdini, Direct	Our Algorithm
100^3	00:24	N/A
200^3	03:44	00:12
400^3	38:00	00:58
800^3	N/A	05:29
1000^3	N/A	11:12

Performance Per Frame

	Houdini, Direct	Our Algorithm
100^3	00:24	N/A
200^3	03:44	00:12
400^3	38:00	00:58
800^3	N/A	05:29
1000^3	N/A	11:12

Performance Per Frame

	Houdini, Direct	Our Algorithm
100^3	00:24	N/A
200^3	03:44	00:12
400^3	38:00	00:58
800^3	N/A	05:29
1000^3	N/A	11:12

Performance Per Frame

	Houdini, Direct	Our Algorithm
100^3	00:24	N/A
200^3	03:44	00:12
400^3	38:00	00:58
800^3	N/A	05:29
1000^3	N/A	11:12

Performance Per Frame

	Houdini, Direct	Our Algorithm
100^3	00:24	N/A
200^3	03:44	00:12
400^3	38:00	00:58
800^3	N/A	05:29
1000^3	N/A	11:12

Performance Per Frame

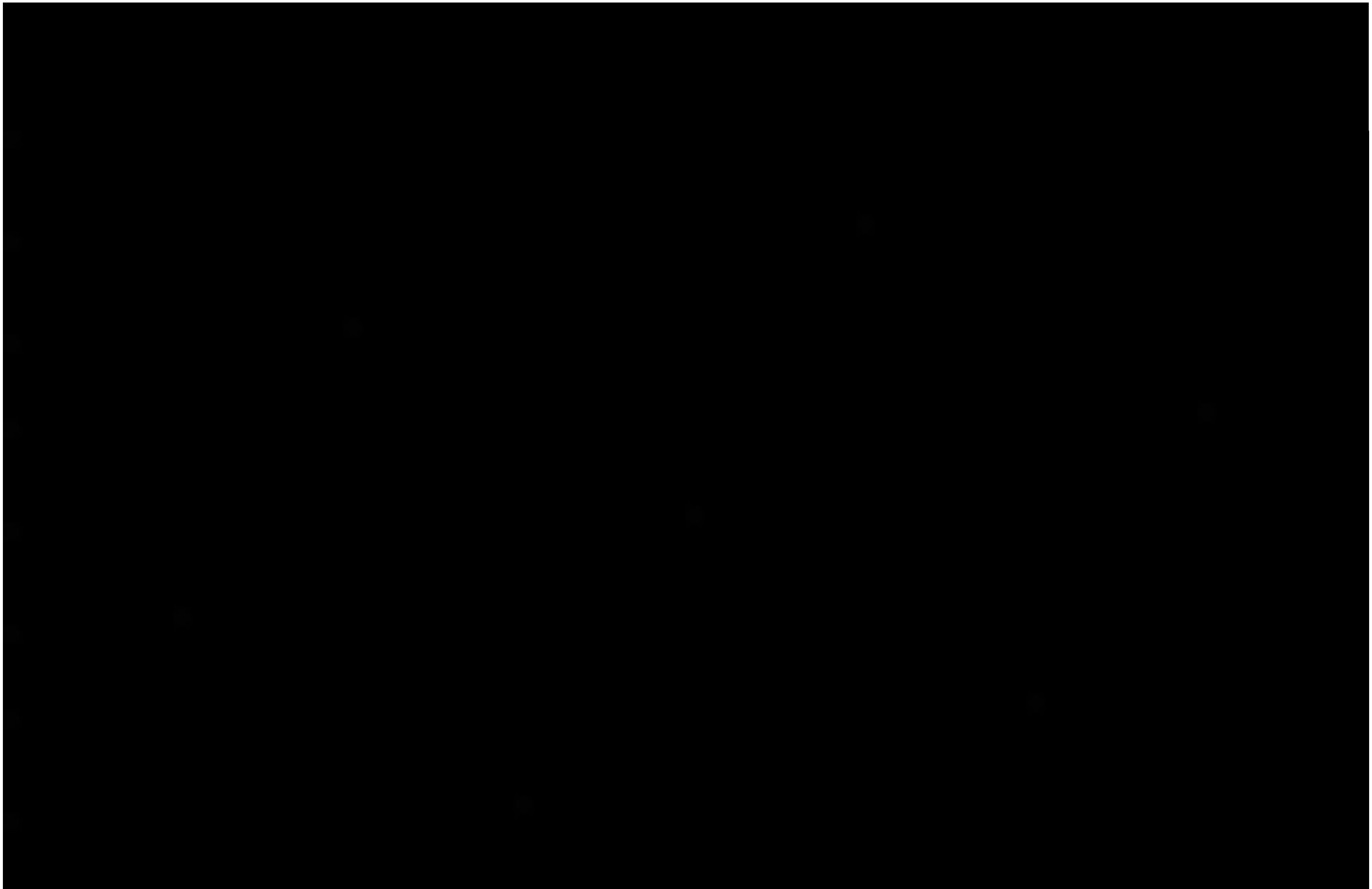
	Dam Break	Pouring
Original Simulation	00:18	01:52
2x Upres	00:06	00:07
4x Upres	00:21	00:34
8x Upres	02:48	03:15

Performance Per Frame

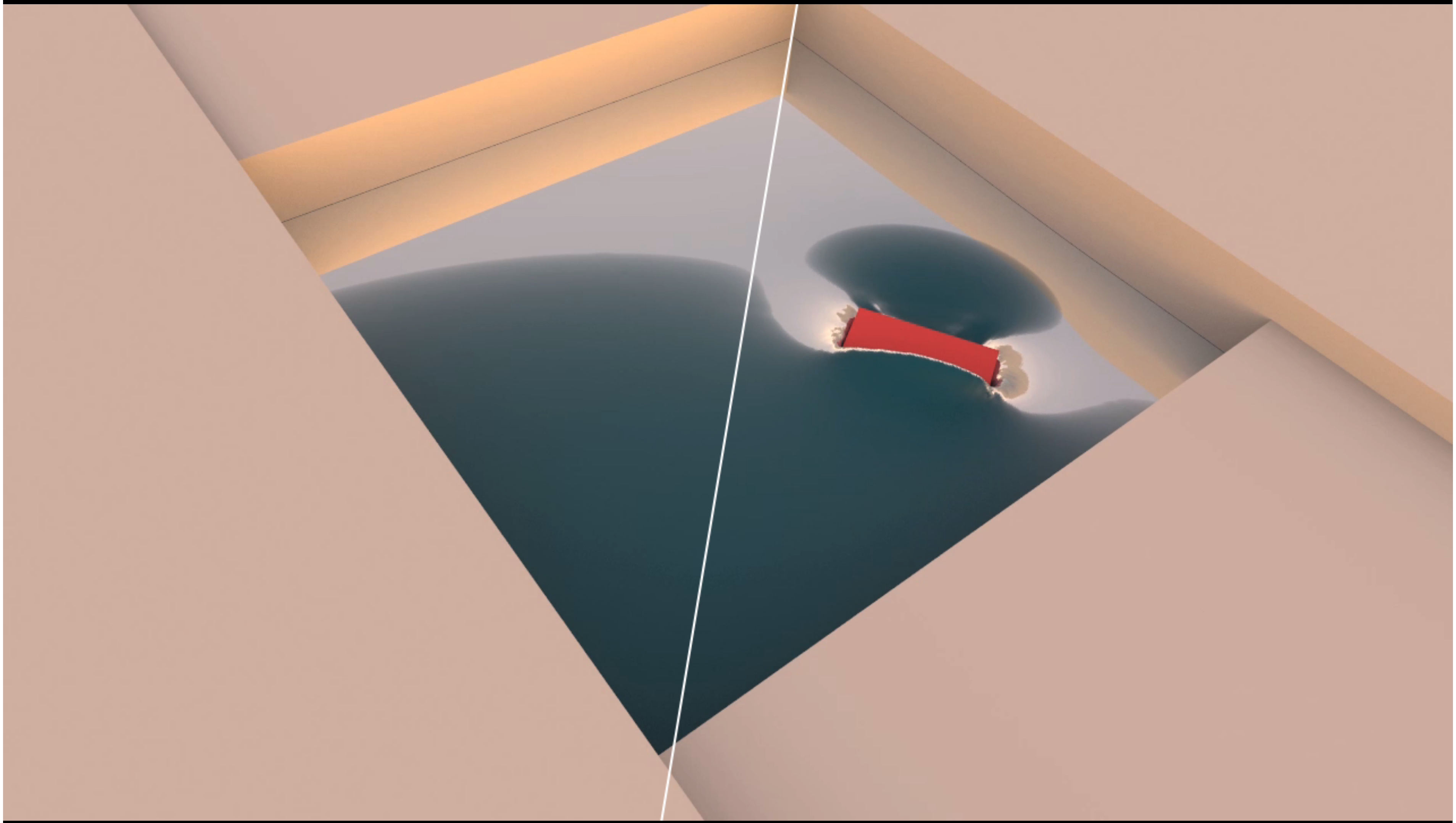
	Dam Break	Pouring
Original Simulation	00:18	01:52
2x Upres	00:06	00:07
4x Upres	00:21	00:34
8x Upres	02:48	03:15

Performance Per Frame

	Dam Break	Pouring
Original Simulation	00:18	01:52
2x Upres	00:06	00:07
4x Upres	00:21	00:34
8x Upres	02:48	03:15

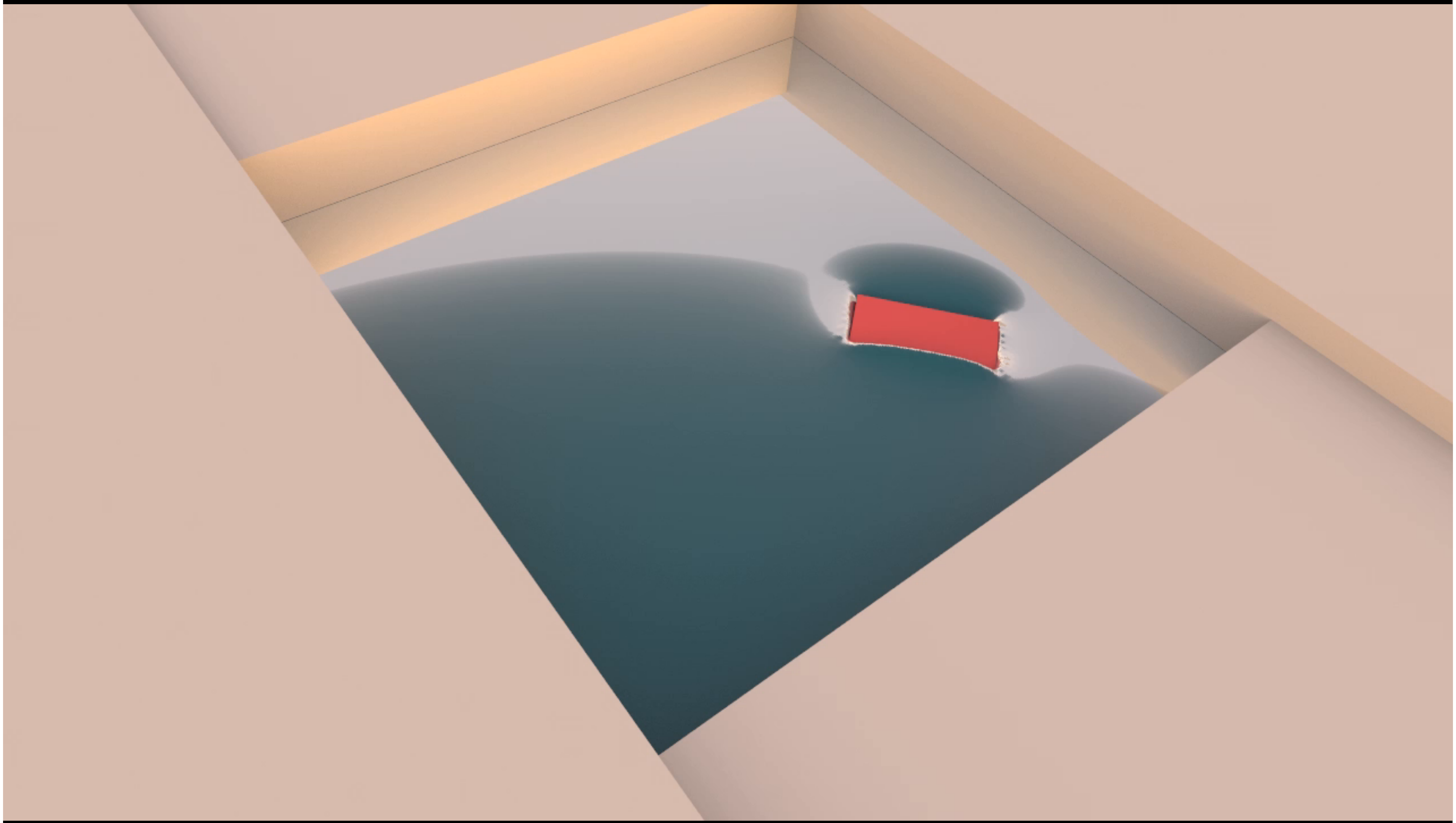


[Nielsen and Bridson 2011]

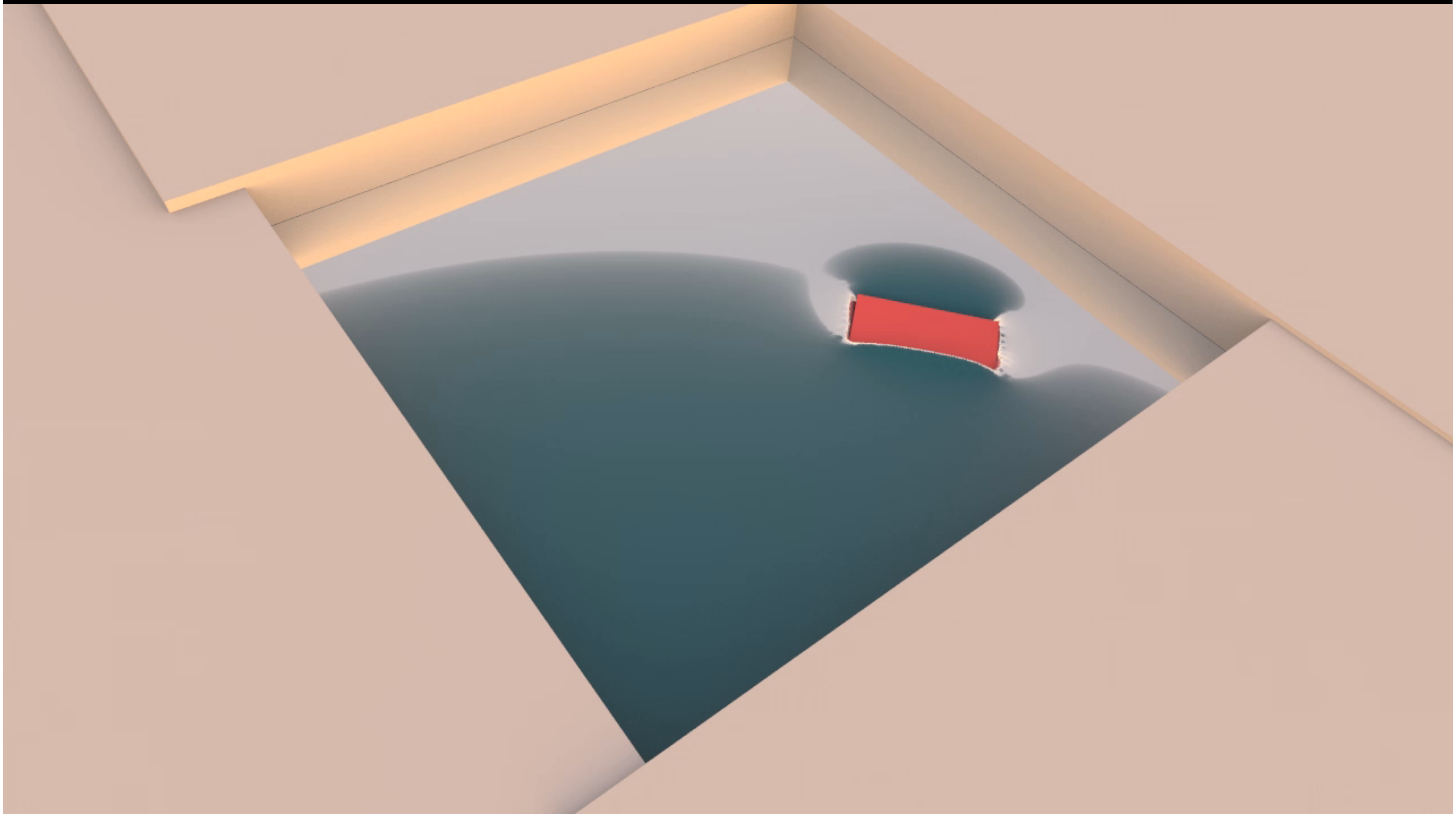


8x up-res

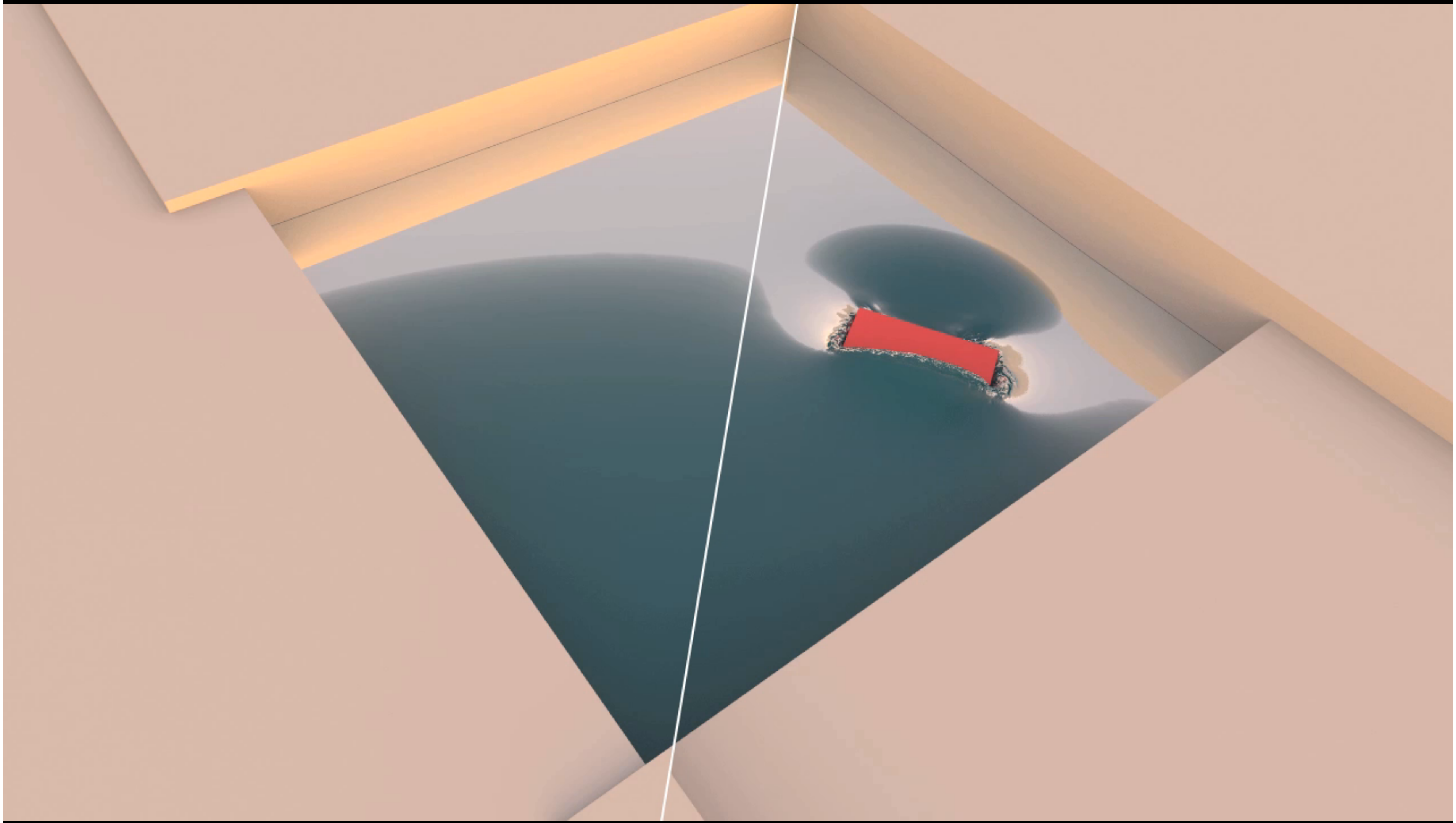
Original



2x up-res, 00:12 per frame



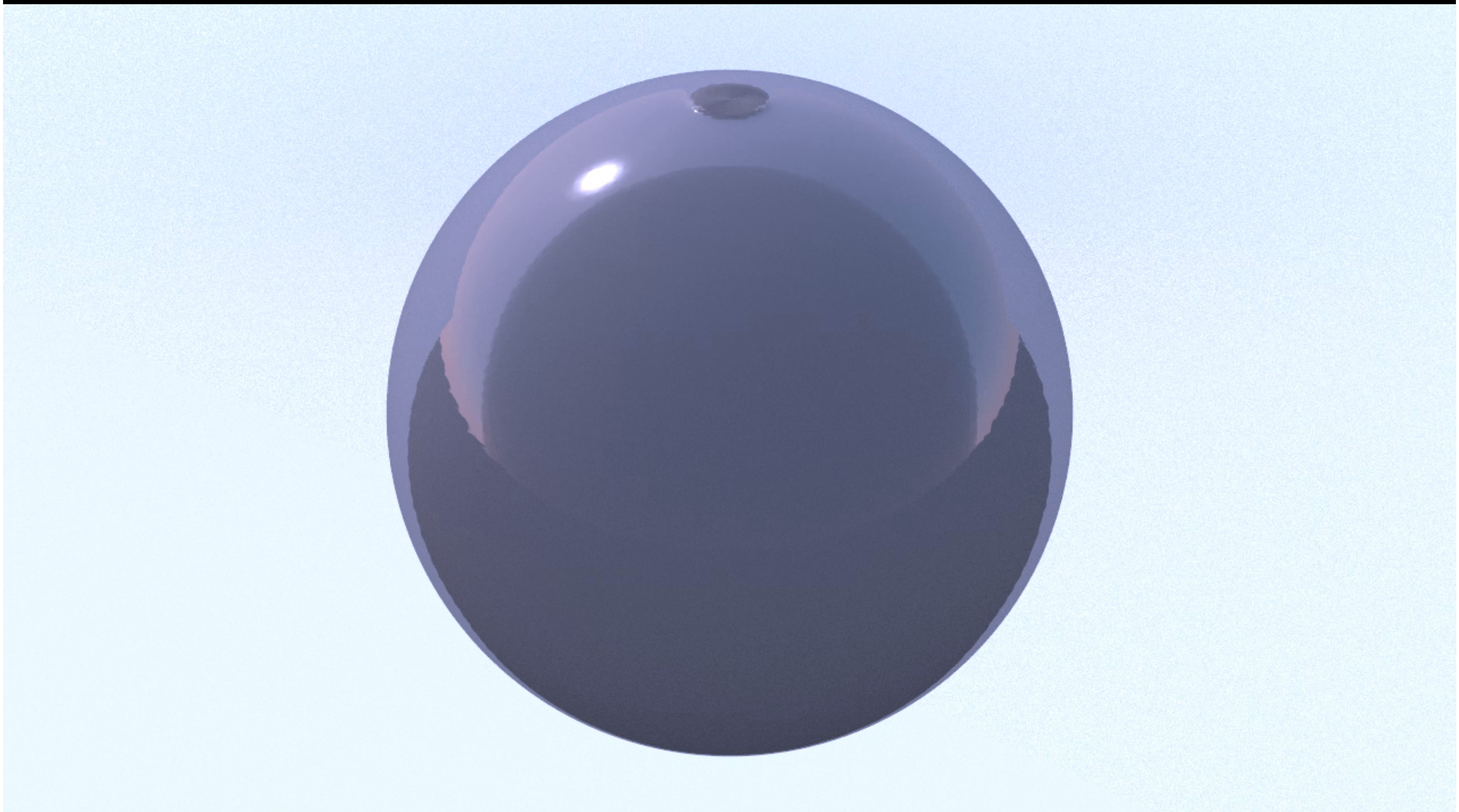
10x up-res, 11:12 per frame



10x up-res

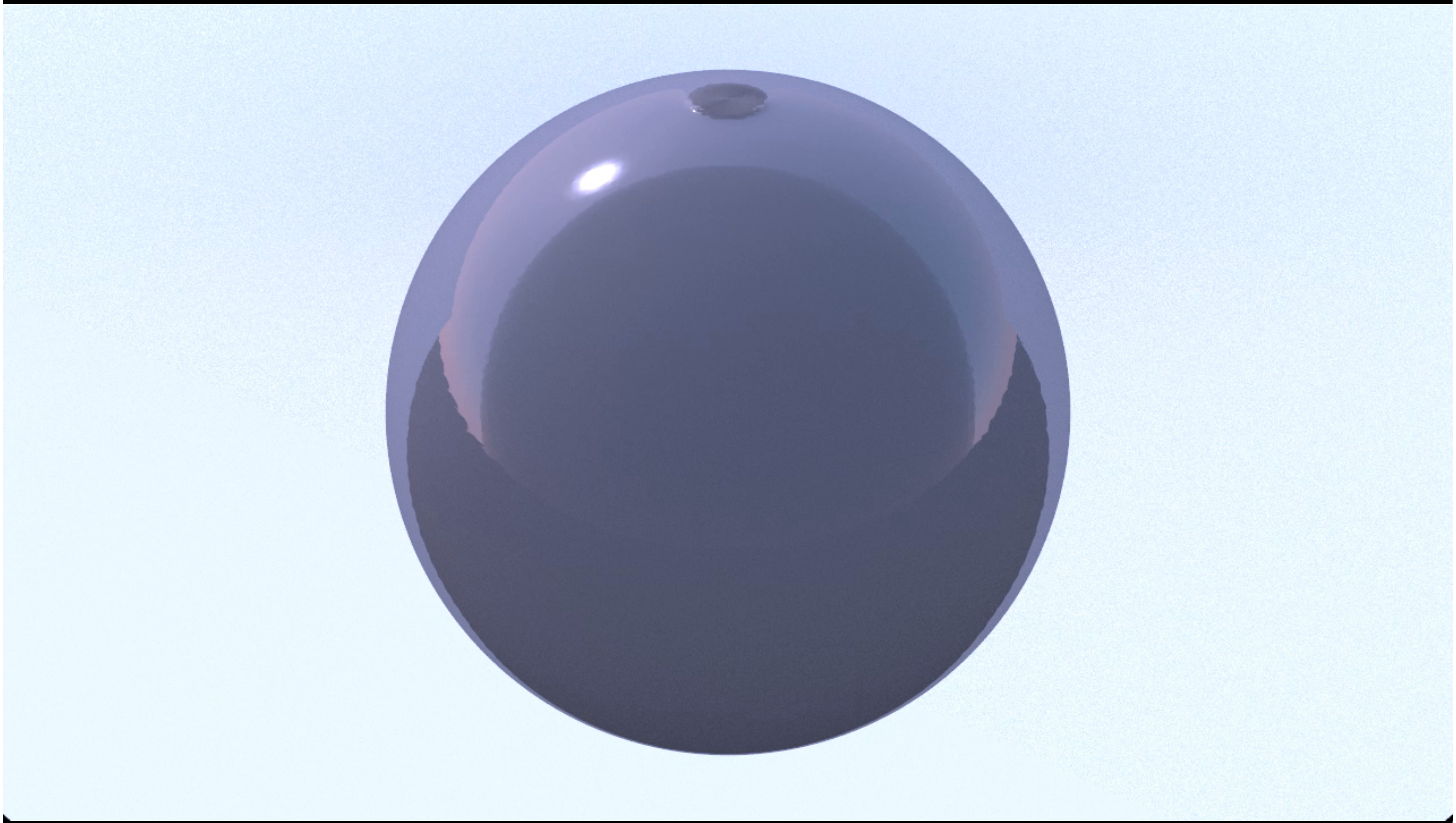
8x up-res

Extension Field



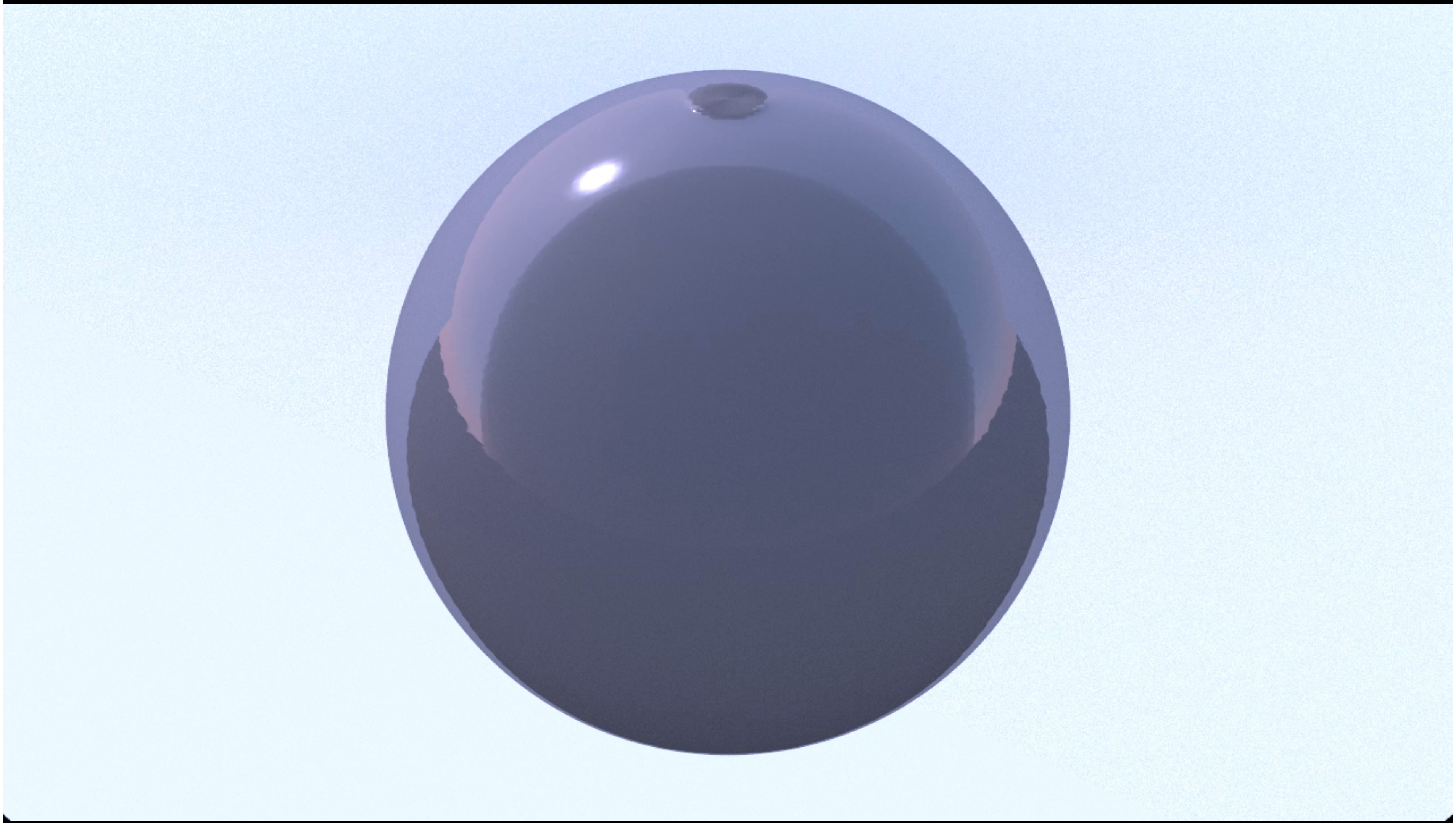
Conventional Extension

Extension Field



Extend once

Extension Field



Frozen core extension