Spatiotemporal Simulation of Biological Processes

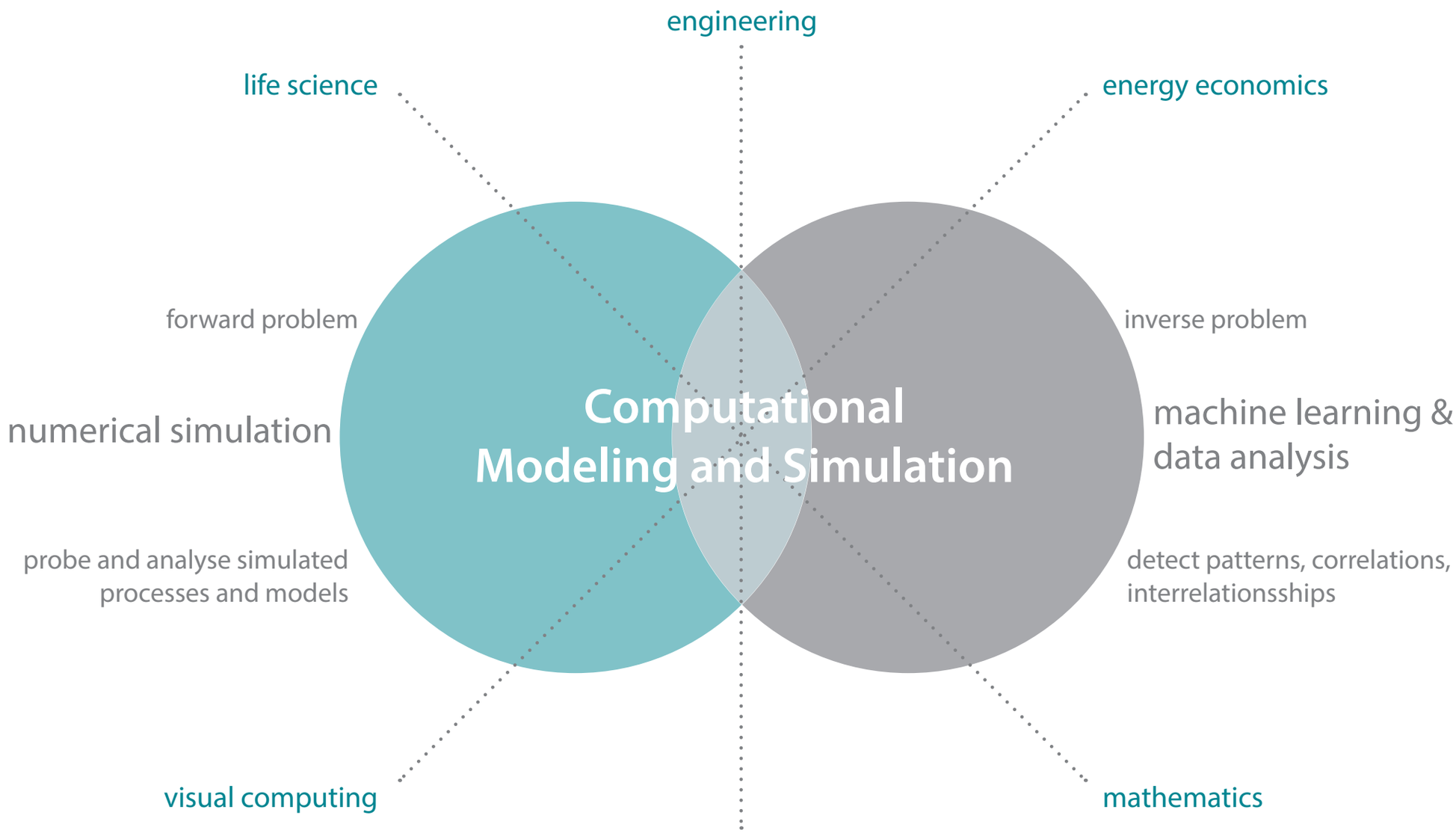
Ivo F. Sbalzarini

MOSAIC Group, CSBD

TUD & MPI-CBG

Studienmöglichkeiten @TUD-INF

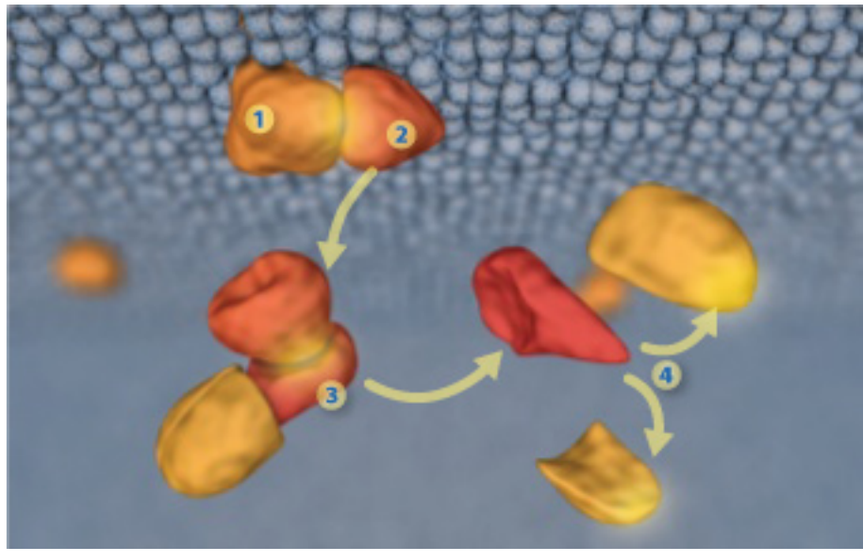
1) Nebenfach "Biowissenschaften"
2) M.Sc. "Computational Modeling and Simulation", Track "Computational Life Sciences"



Biology Computes

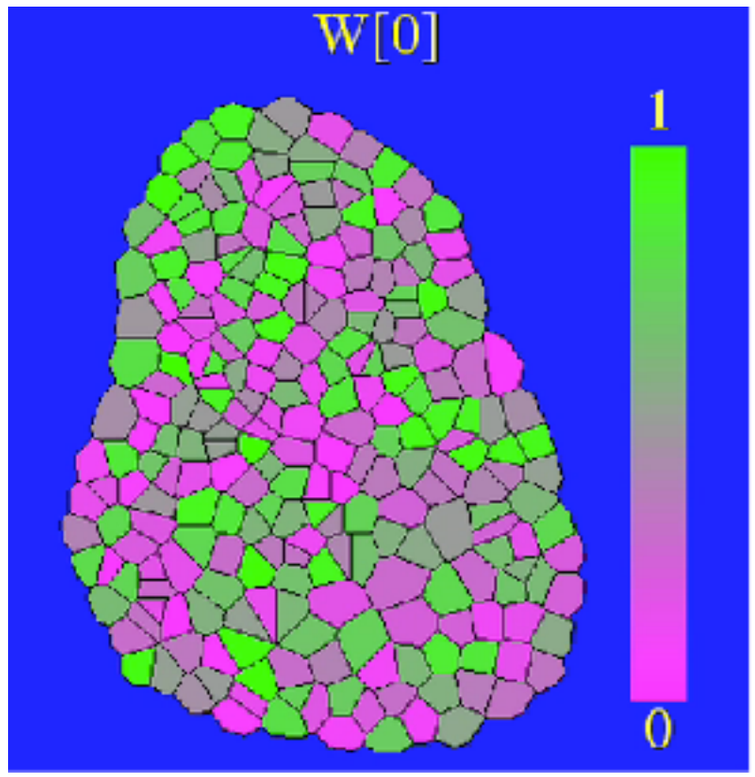
Cells execute **programs**:

- Genetic programs
- Communication programs
- Decision making
- Signaling networks
- Endosomal sorting
- …



Cells **communicate** by:

- chemical signals
- mechanical signals
- cell-cell contacts
- motion
- …



Meyerowitz, Caltech

Systems Biology

It's not about **using** computers to process biological data or to simulate biophysical models,

but

to understand the biological system itself as a **computing process**, reverse-engineer its "algorithms" and its "grammar",
and be able to reprogram (re-engineer) it.

biological processes are inherently *algorithms* that nature has designed to solve computational problems.

algorithmsinnature.org

Computing in Biology

> Computer science will be for 2020 biology what mathematics is for today's physics.

Microsoft Report "2020 Science"

Where and Why Computing in Biology?

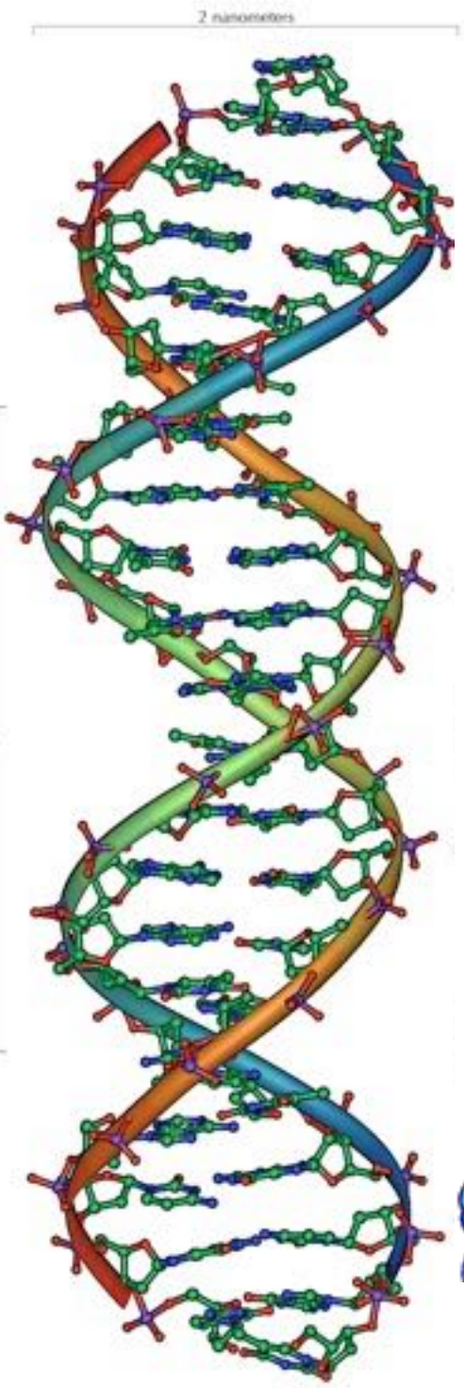
Amount of data / Reproducibility	→	Manual evaluation too slow or too unreliable
Complexity	→	System behavior not apparent from description
Time/length scales	→	Too big/small, slow/fast for experimental measurement
Ethics	→	No living beings involved
Controllability	→	Variables controllable
Observability	→	Variables measurable

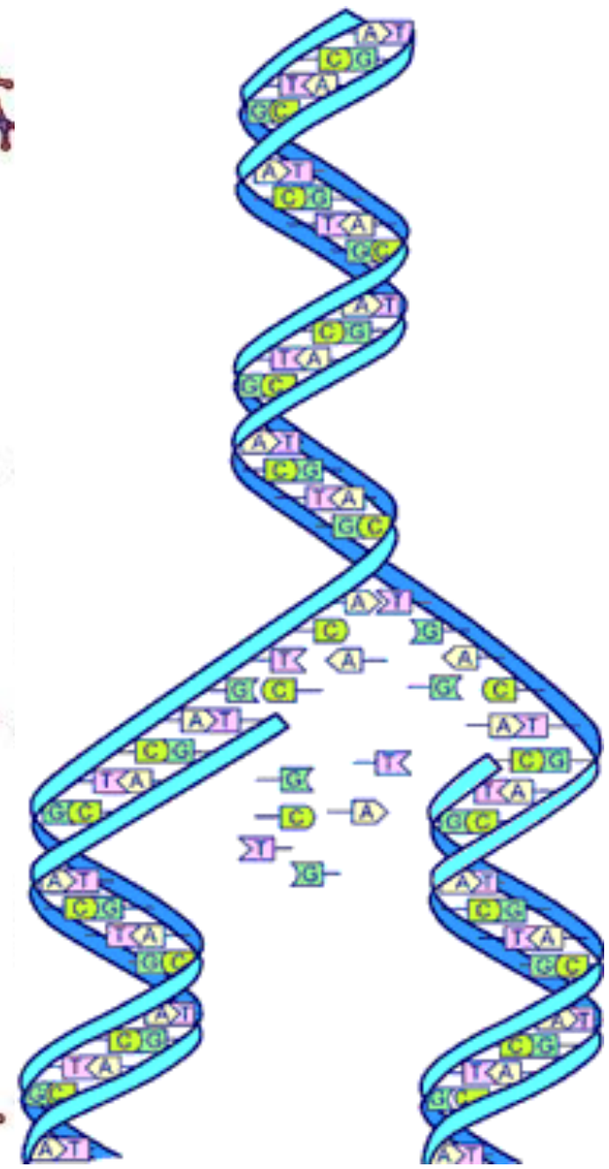
Computational Challenges in Biological Systems

- Hierarchical
- Regulated
- Complex shapes
- Non-equilibrium
- Nonlinear
- Coupled
- Plastic

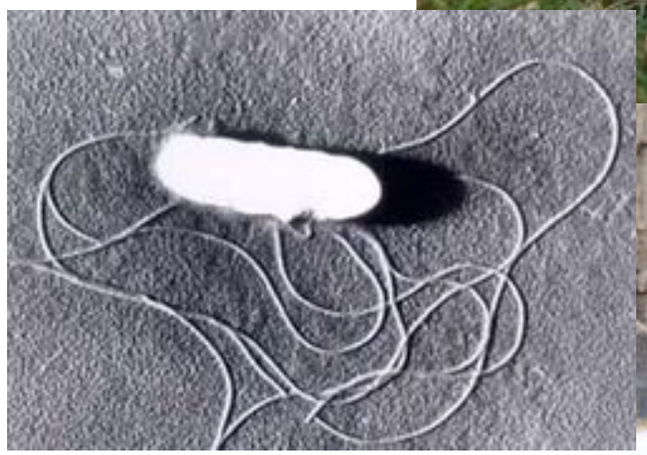
Goal: System Dynamics

Source Code??





???



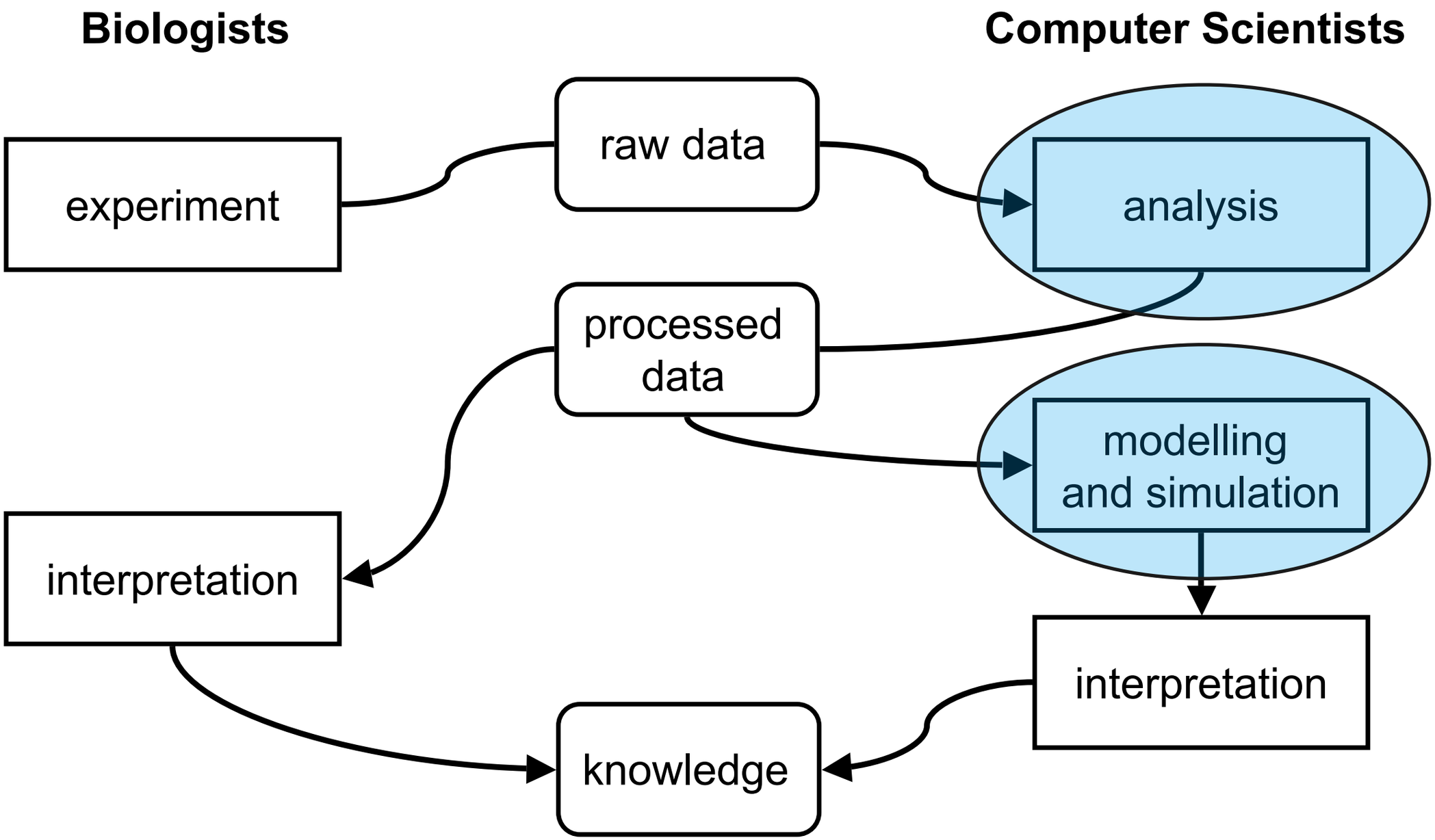
Bilder: Wikipedia

Some Goals

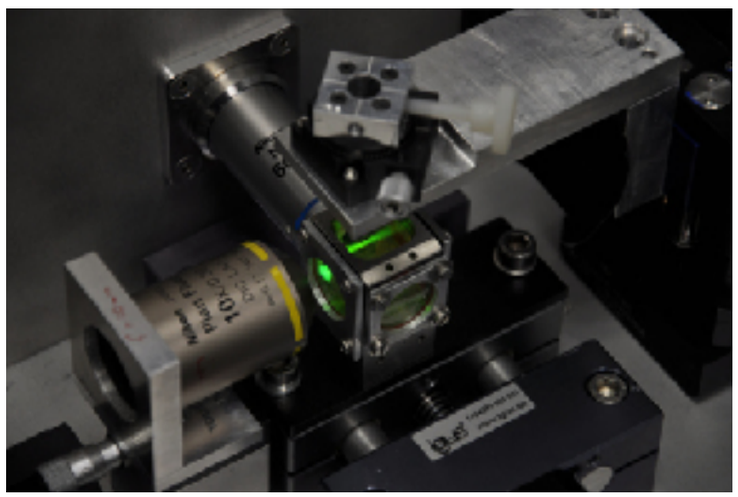
In order to understand the mechanistic working of biological systems, we also have to consider:

- Spatial organization and compartmentalization
- Temporal plasticity and dynamics
- Environmental influences
- Physics of interactions
- Regulatory mechanisms

A Typical Workflow in Systems Biology



What we want to do…

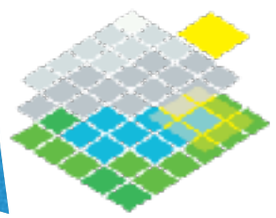


PPM_RC

Real-time

analysis

Real-time visualization



scenery

Simulation

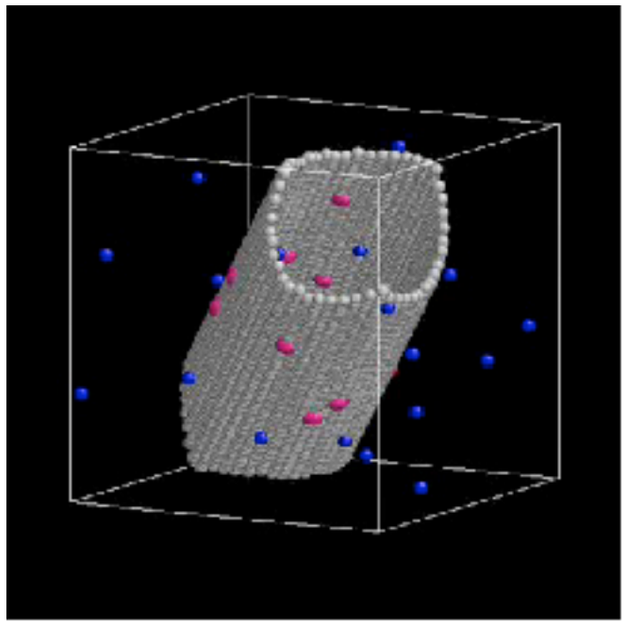
OpenFPM

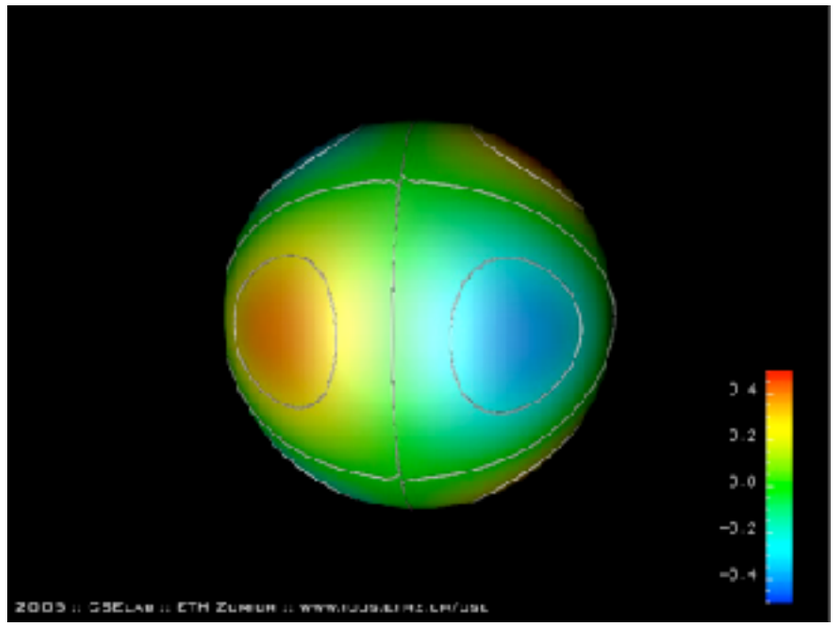
Photo: SZonline

Particles: a Unifying Computational Framework

- Particle methods allow seamless treatment of continuous and discrete systems, both stochastically and deterministically
- In discrete systems, particles often correspond to real-world entities (atoms in molecular dynamics, cars in traffic simulations)
- In continuous systems, particles represent discretization points (fluid elements in flow simulations, mass in diffusion, etc.)

Examples:





Particles: a Unifying Computational Framework

Particles are computational elements, defined by their position and an arbitrary set of properties they carry.

discrete	continuous
• Particles = real-world objects • Particle-particle interactions model physical (biological, financial, etc.) interactions • Macroscale mathematical models are not required • E.g.: molecular dynamics, agent-based models, Monte Carlo methods, etc.	• Particles = Lagrangian tracer points • Particle-particle interactions are designed such that the collective dynamics of the particles approximates a differential operator • E.g.: Vortex Methods, SPH, PSE, etc.

Particle-particle interactions can be deterministic or stochastic

Talk Outline

Deterministic particle methods for multi-resolution spatiotemporal simulations

Particle-based bio-image processing

A parallel middleware and domain-specific language for particle-mesh methods on heterogeneous parallel computers

Talk Outline

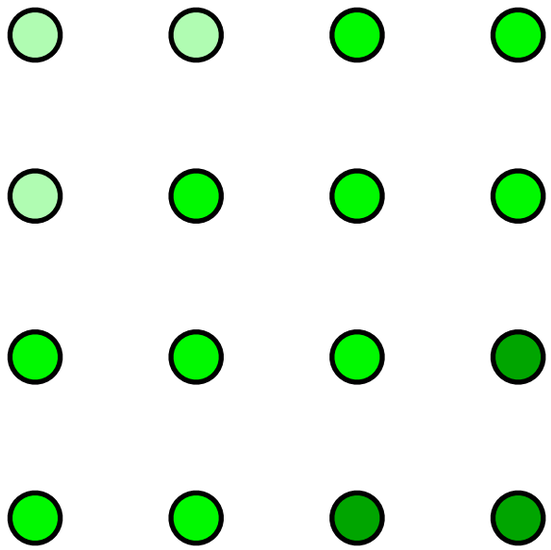
Deterministic particle methods for multi-resolution spatiotemporal simulations

Particle-based bio-image processing

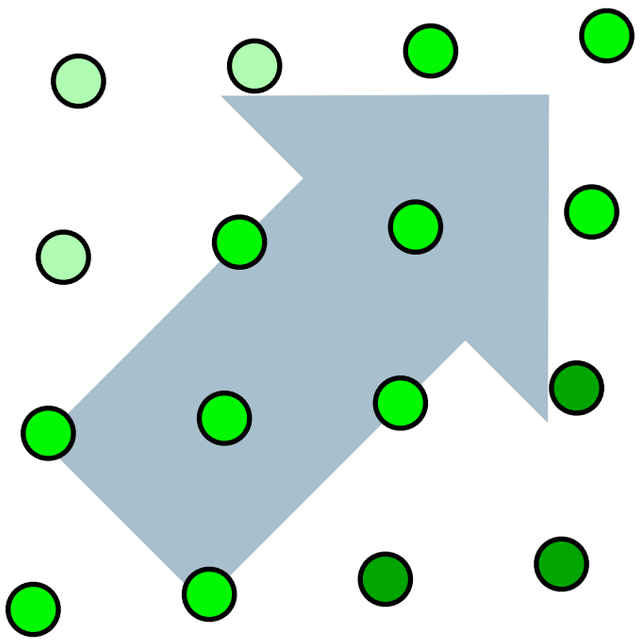
A parallel middleware and domain-specific language for particle-mesh methods on heterogeneous parallel computers

Continuum Particle Methods

transport equation: $\dfrac{\partial f}{\partial t} + \nabla \cdot (f\mathbf{u}) = g$



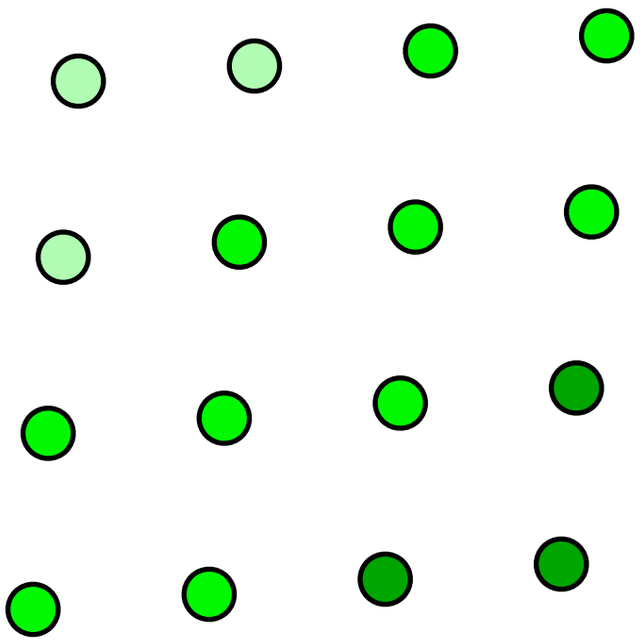
position $\mathbf{x}_p$
volume v_p
strength ω_p



convect particles

$$\frac{\mathrm{d}\mathbf{x}_p}{\mathrm{d}t} = \mathbf{u}$$

$$\frac{\mathrm{d}v_p}{\mathrm{d}t} = v_p \nabla \cdot \mathbf{u}$$



evolve particle strengths

$$\frac{\mathrm{d}\omega_p}{\mathrm{d}t} = v_p g$$

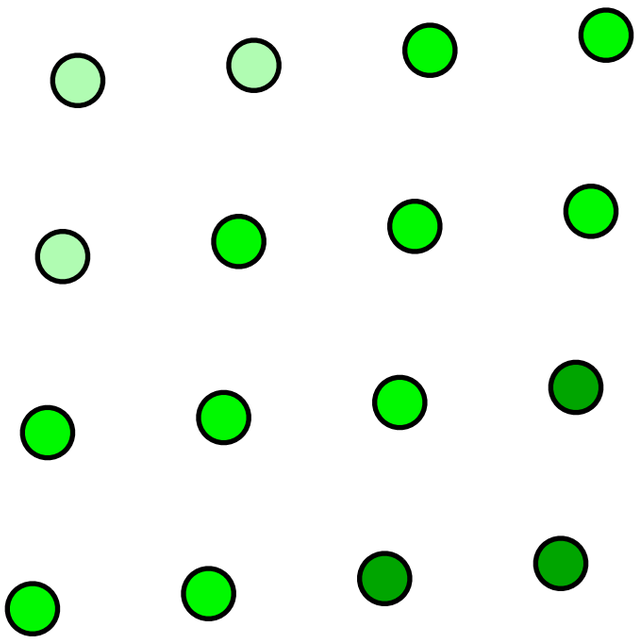
Differential Operators over Particles

transport equation: $\frac{\partial f}{\partial t} + \nabla \cdot (f\mathbf{u}) = g$

source term g often depends on derivatives of the field f, e.g.

$$g = \nabla \cdot (\nu \nabla f)$$

if f is diffused.



evolve particle strengths

$$\frac{\mathrm{d}\omega_p}{\mathrm{d}t} = v_p g$$

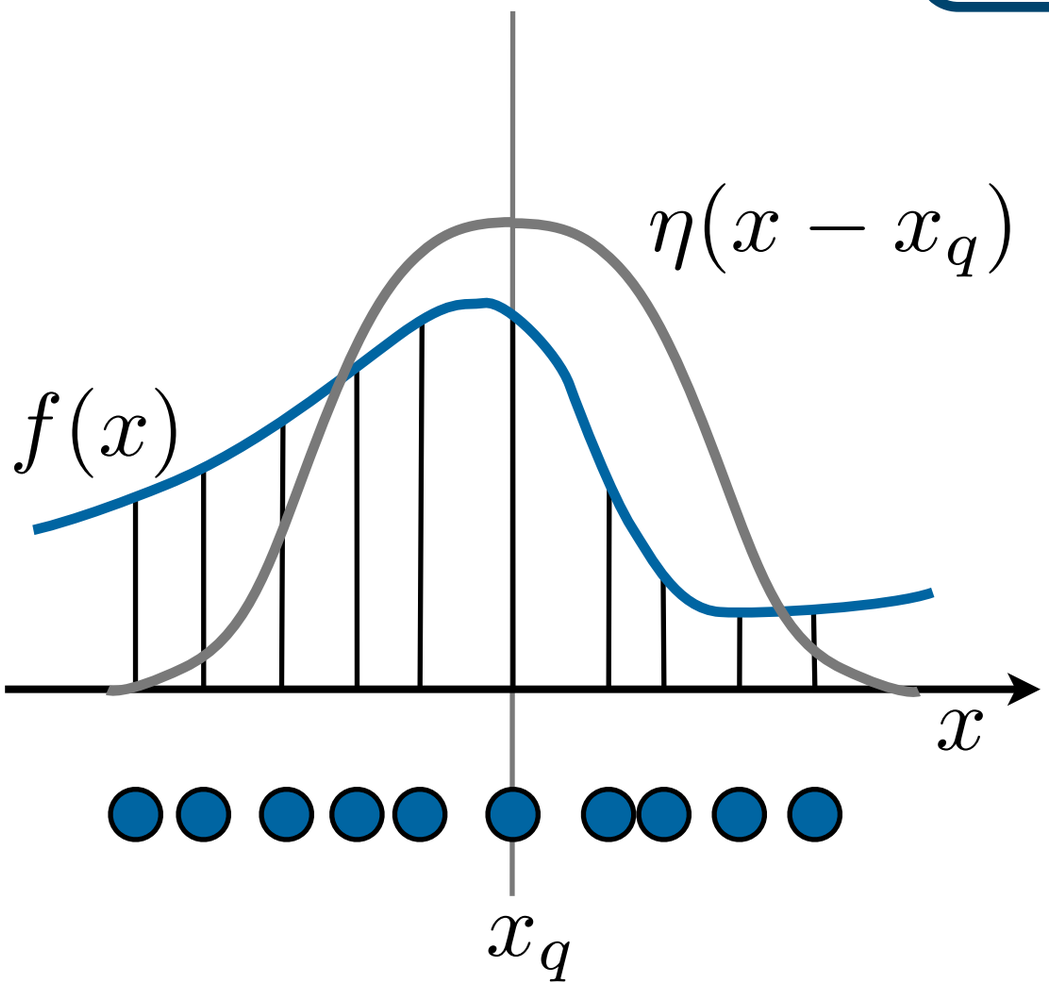
evaluation of derivatives of f is crucial

The PSE Operator Approximation

moment conditions on kernel function η

$$\left.\frac{\partial^\beta f}{\partial x^\beta}\right|_{x_q} \approx \int_{-\infty}^{\infty} [f(x_q) \pm f(x)]\eta(x - x_q)dx$$

$$\approx \sum_p v_p [f(x_q) \pm f(x_p)]\eta(x_p - x_q)$$



Degond & Mas-Gallic, The weighted particle method for convection-diffusion equations. Part 1: The case of isotropic viscosity, Math. Comp. 53 (188) (1989) 485-507

Eldredge et al., A general deterministic treatment of derivatives in particle methods, J. Comput. Phys. 180 (2002) 686-709

$$\eta \rightarrow \eta^q$$

Schrader et al., J. Comput. Phys. 229:4159–4182, 2010.

The Endoplasmic Reticulum (ER)

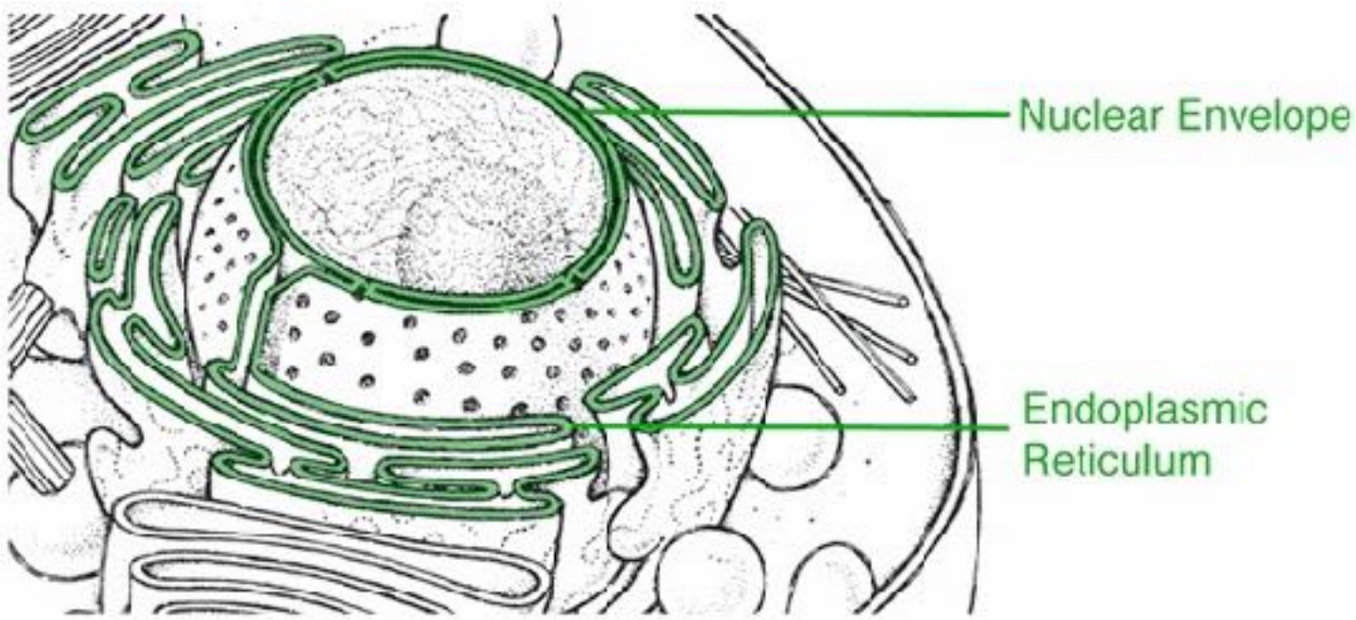
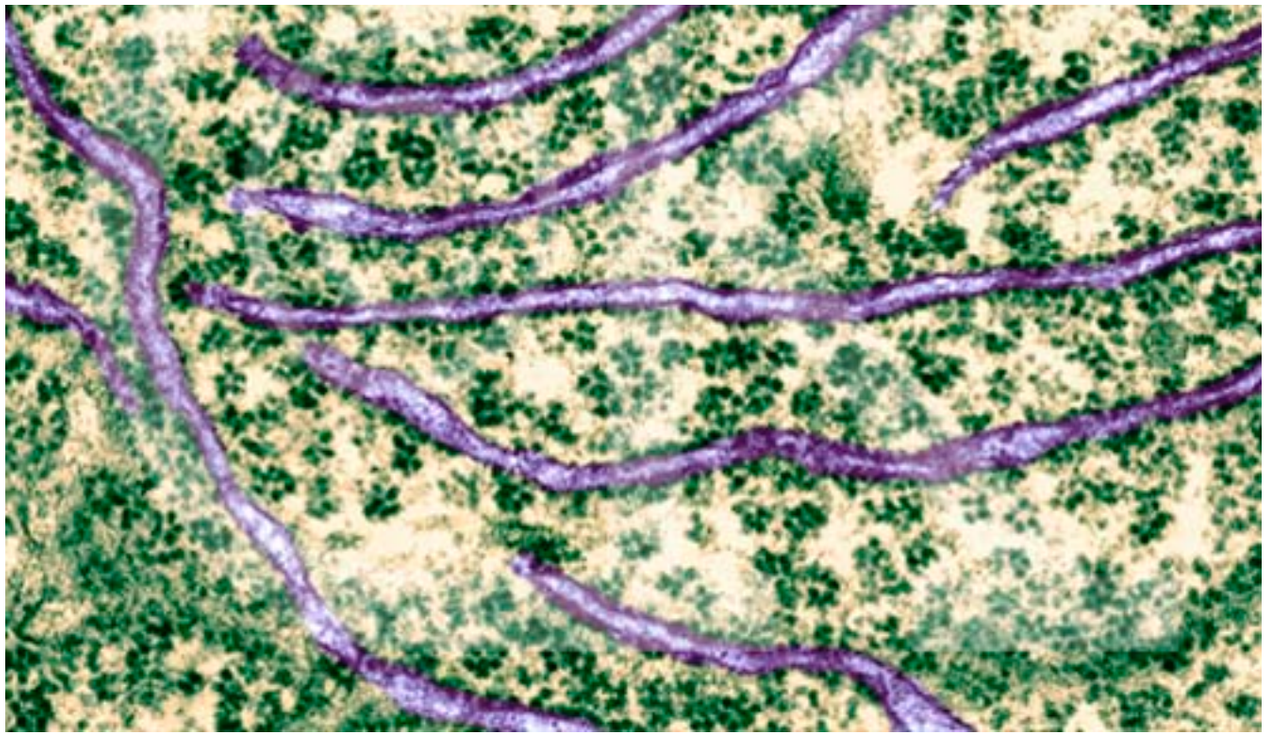


Figure: De Duve, Une visite guidée de la cellule vivante, 1987.



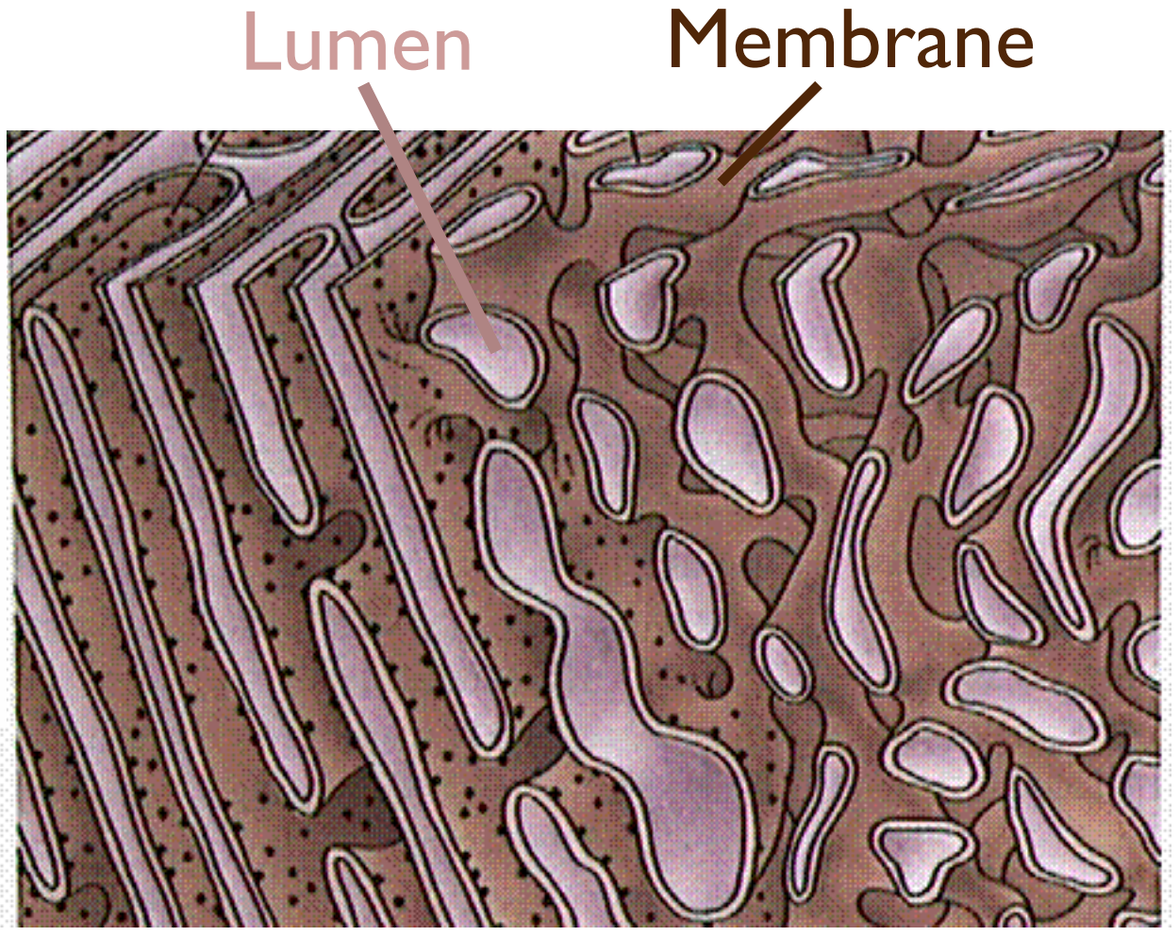


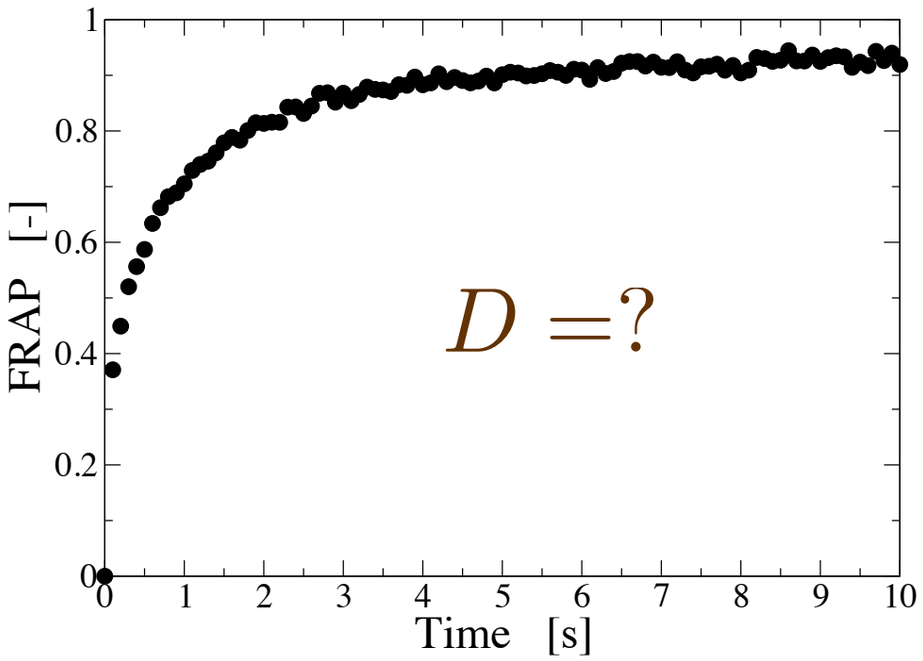
Figure: Purves et al., Life: The Science of Biology, W.H. Freeman.

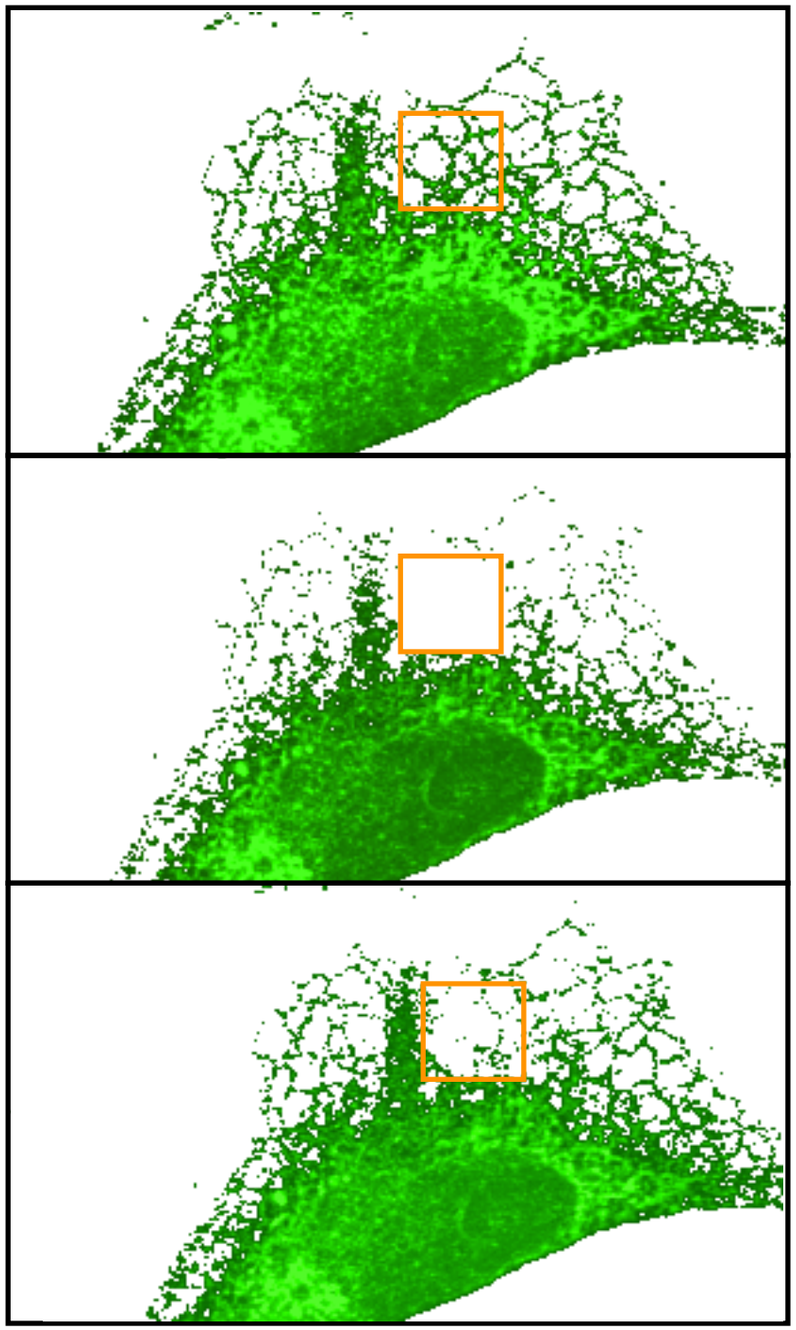
The main biosynthetic organelle in Eukaryotes: Protein and lipid synthesis.

Enclosed by a contiguous membrane

FRAP: Fluorescence Recovery After Photobleaching

- Tag protein fluorescently
- Bleach **region of interest** with Laser
- Monitor influx of unbleached protein





pre-bleach

t=0

t=2 min

Multi-Scale Issue in FRAP

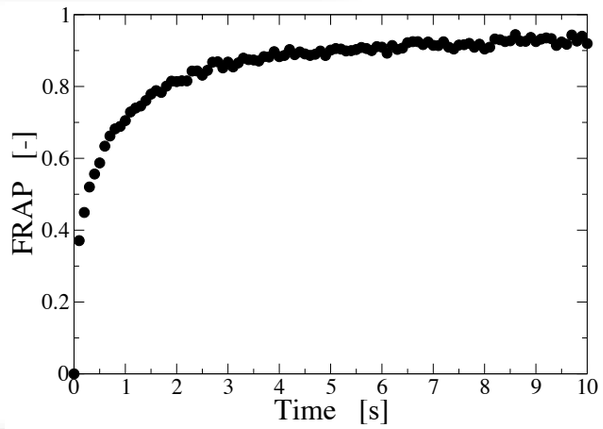
- **Discrepancy of scales:** Observation of an integral quantity, averaged over Region Of Interest (ROI)
- ROI is larger than geometrical features of ER
- 2D projection or slice of the 3D object

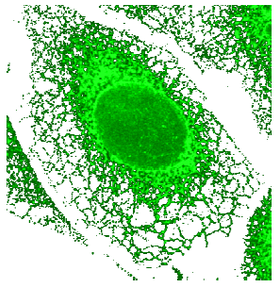
Model needed to control the artifacts of the method and be able to correct for them!

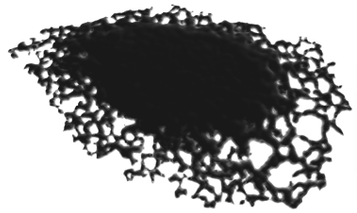
A Computational Solution

FRAP experiment

Experimental FRAP curve







Micrograph sections of the same organelle

3D reconstruction

D_{sim}

Simulation

Simulated FRAP curve

Fit in time

Time unit t_s

Molecular diffusion constant

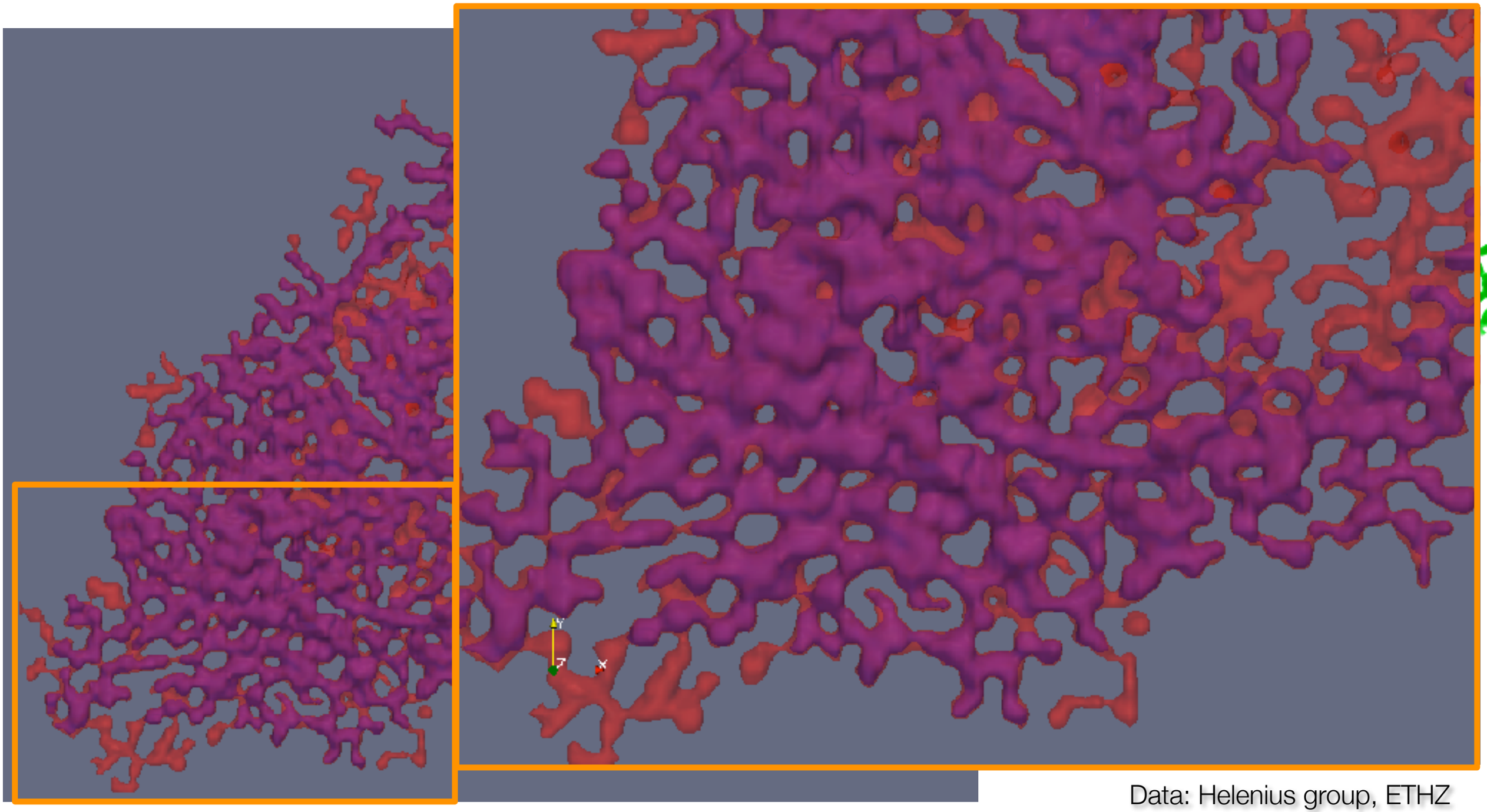
$$D = \frac{D_{\mathrm{sim}}}{t_s}$$

Example Images

ss-GFP-KDEL
VERO cells
confocal microscope
3D PSF measured

Data: Helenius group, ETHZ

3D Deconvolving Segmentation



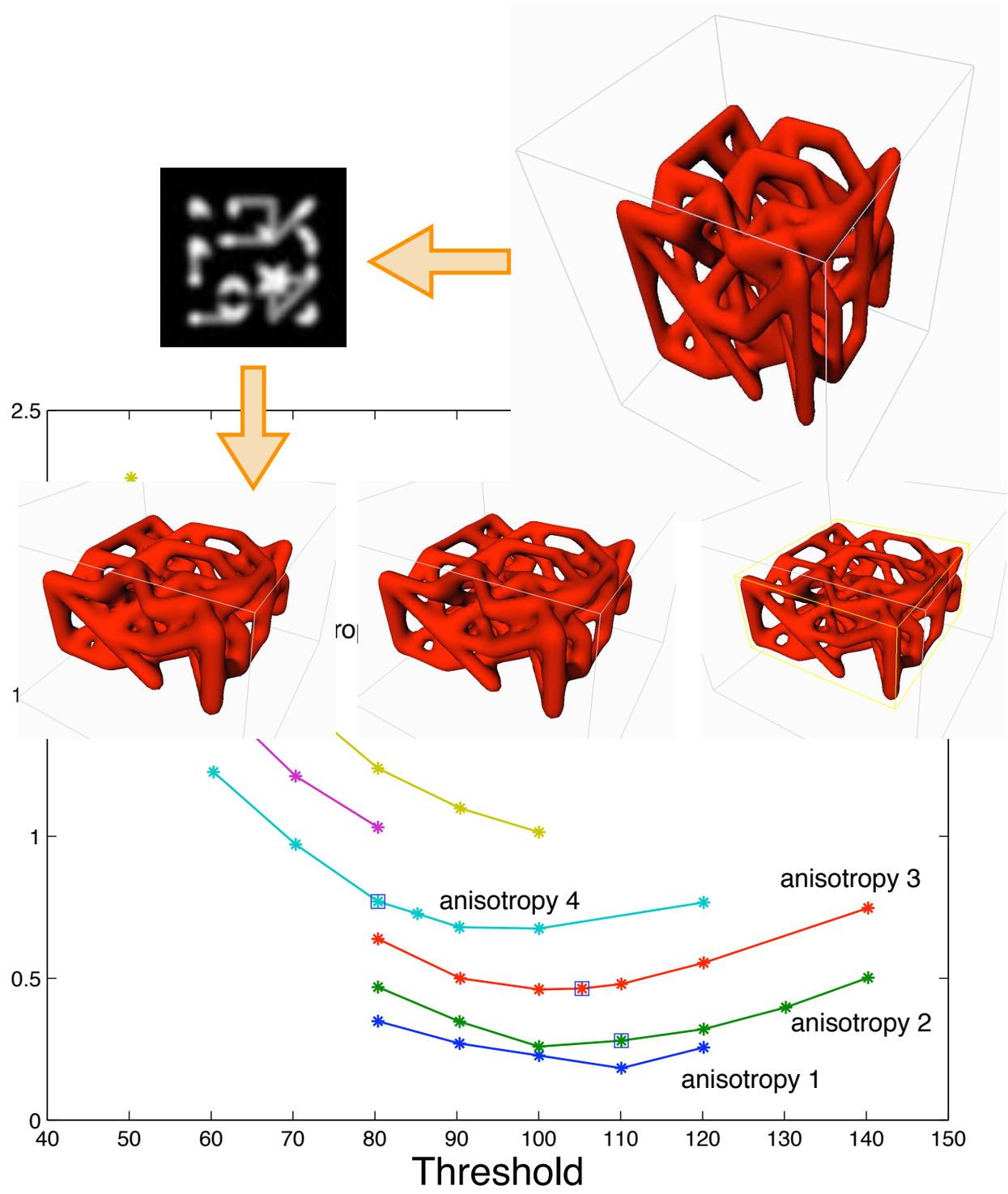
Data: Helenius group, ETHZ

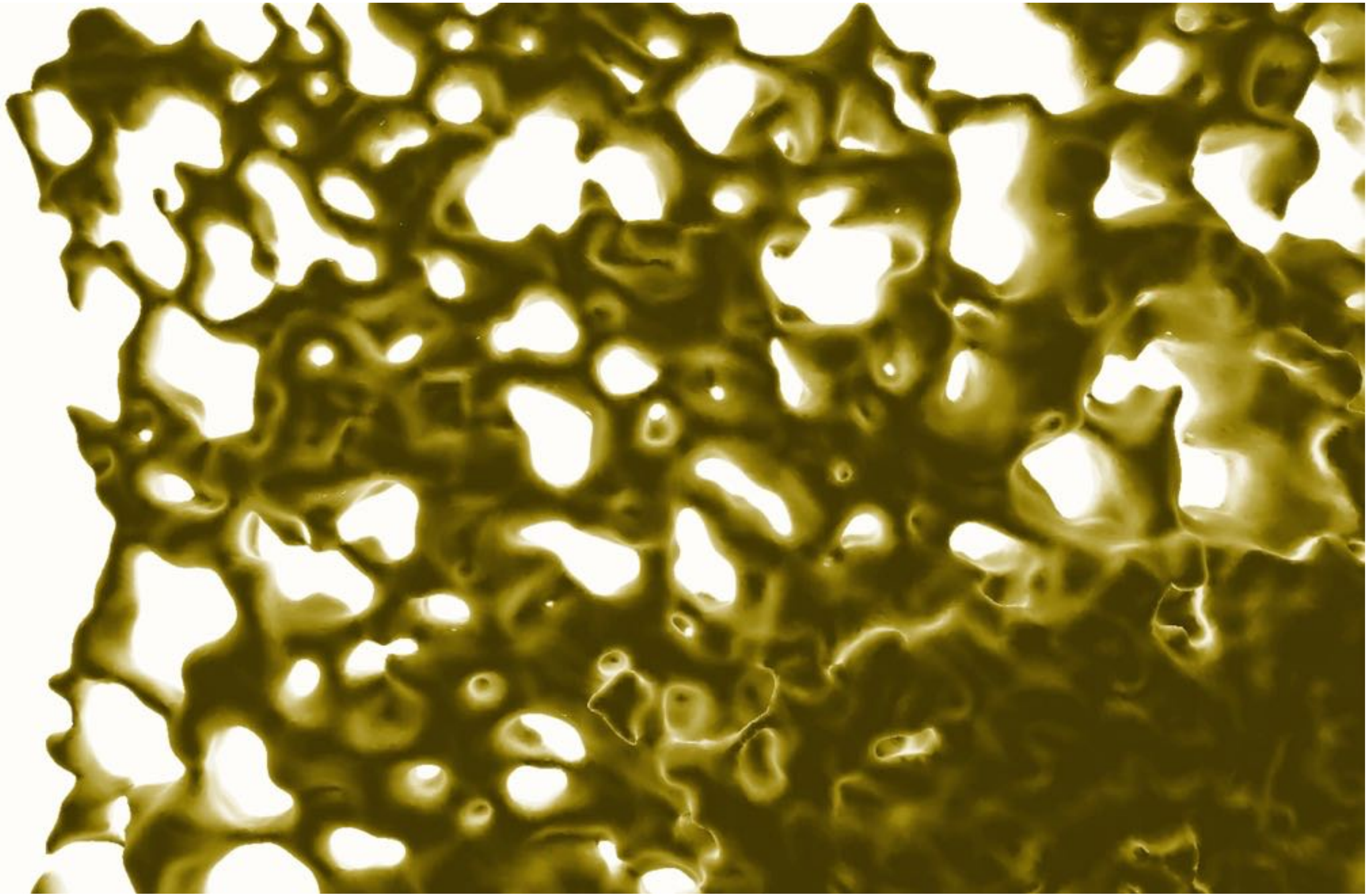
3D Reconstruction error

Reconstruction error
determined by:

- Threshold
- Resolution of the microscope

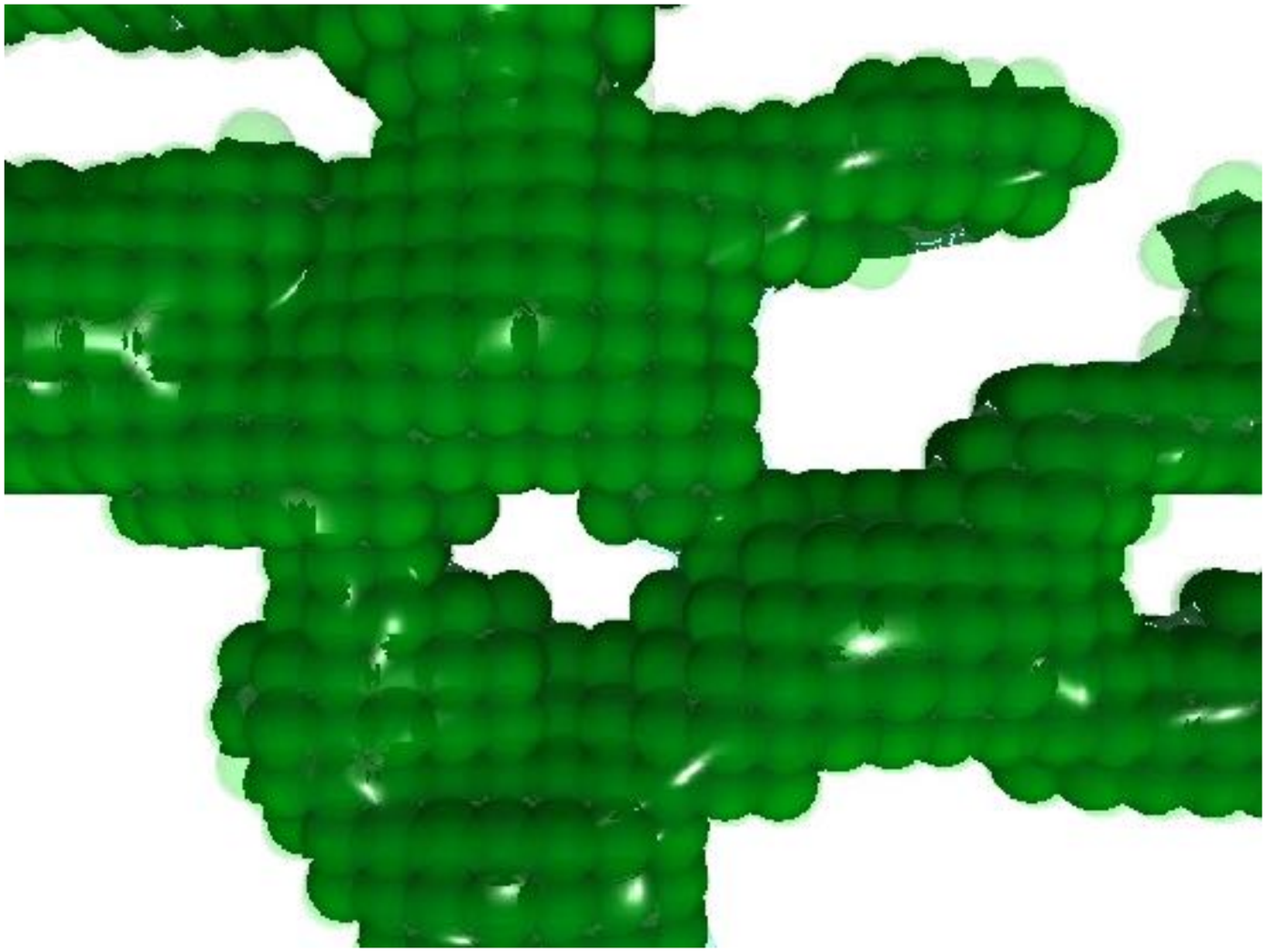
Find optimal threshold in the computer





Particle Method for Diffusion Simulations

Computational elements: **Particles** $(\boldsymbol{x}_p,\, V_p,\, \boldsymbol{f}_p),\, p = 1, \ldots N$
at locations $\boldsymbol{x}_p$ with volumes V_p and **property vector $\boldsymbol{f}_p$:**



$$\boldsymbol{f}_p = (u_p, \boldsymbol{x}_p, \ldots)$$

Diffusion:

Particles exchange MASS with their neighbors.

Degond & Mas-Gallic, Math. Comput. 53:509. 1989.

Diffusion in space

Formulation of the diffusion operator on particles

For (anisotropic) diffusion:

$$\frac{du_p}{dt} = \sum_{q \neq p} \sigma_\epsilon(\boldsymbol{x}_p, \boldsymbol{x}_q, \boldsymbol{D}, t)(u_q - u_p)V_q$$

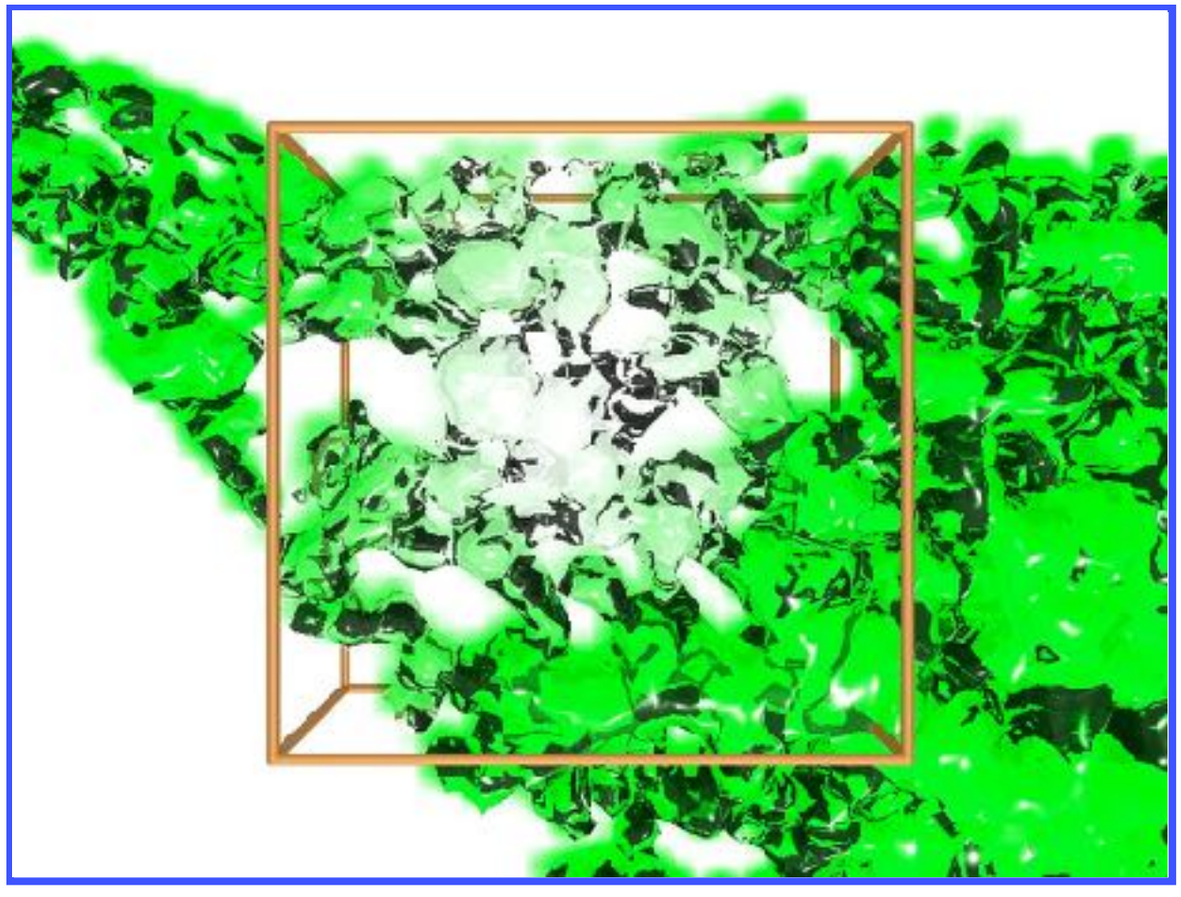
Degond & Mas-Gallic, Math. Comput. 53:509. 1989.

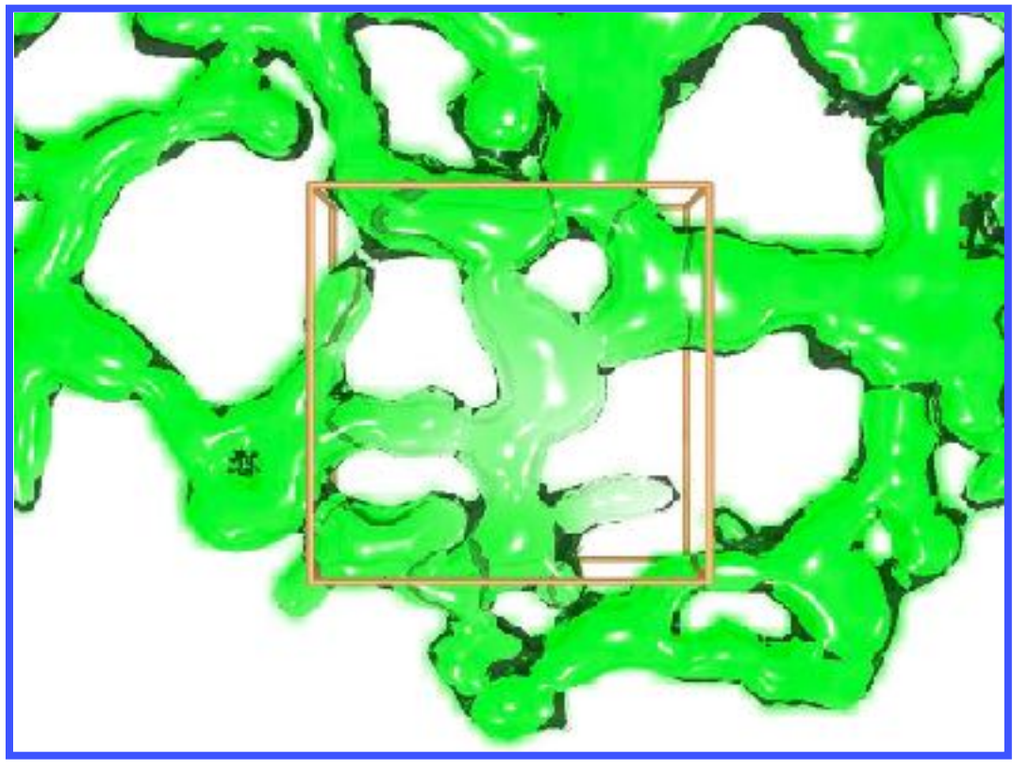
Extension to any differential operator:

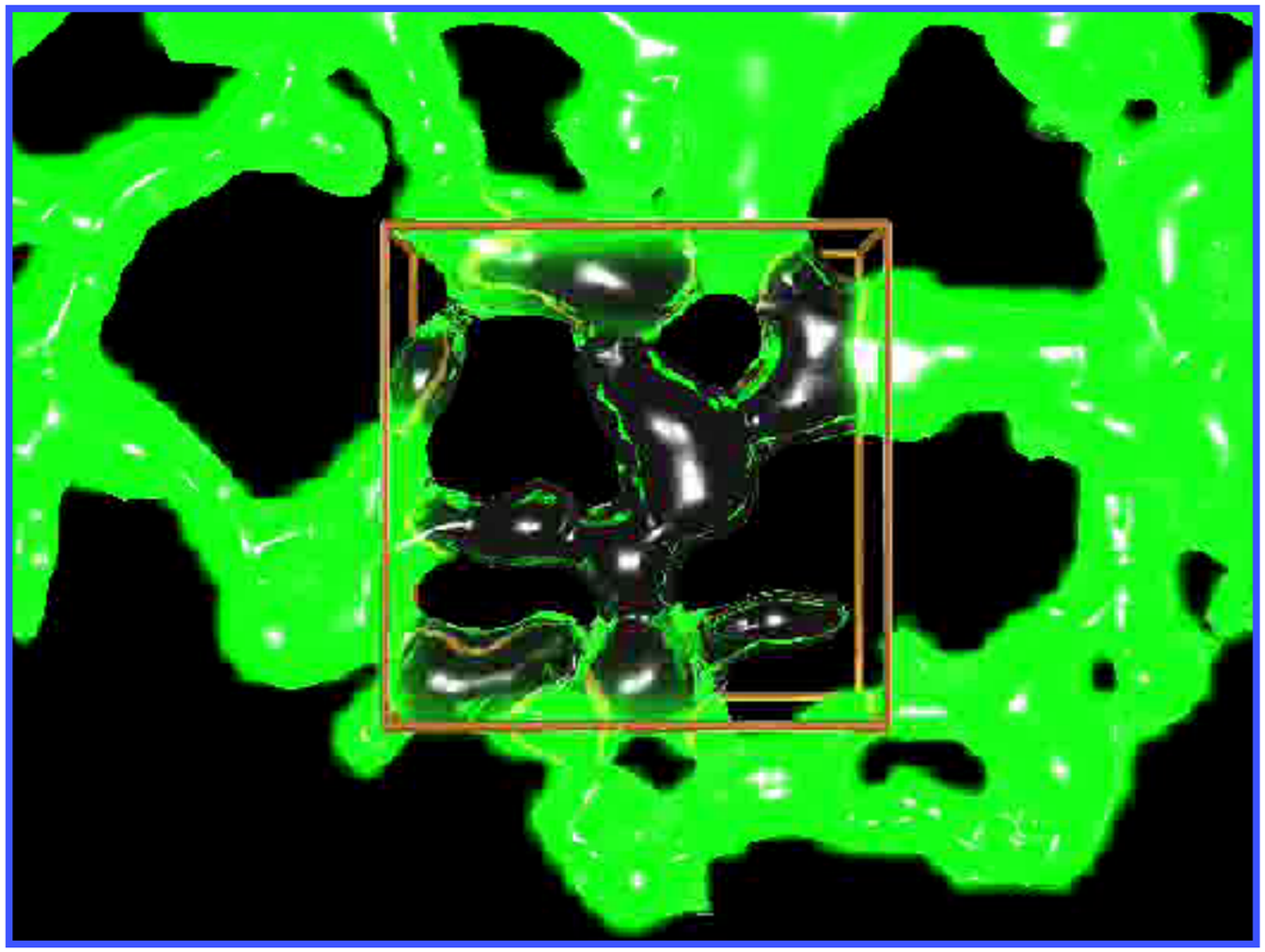
$$L_h^\beta f(\boldsymbol{x}_p) \approx \frac{1}{\epsilon^{|\beta|}} \sum_q V_q \left(f(\boldsymbol{x}_q) \pm f(\boldsymbol{x}_p)\right) \eta_\epsilon^\beta(\boldsymbol{x}_p - \boldsymbol{x}_q)$$

Eldredge et al., J. Comp. Phys. 180:686. 2002.

Lumen

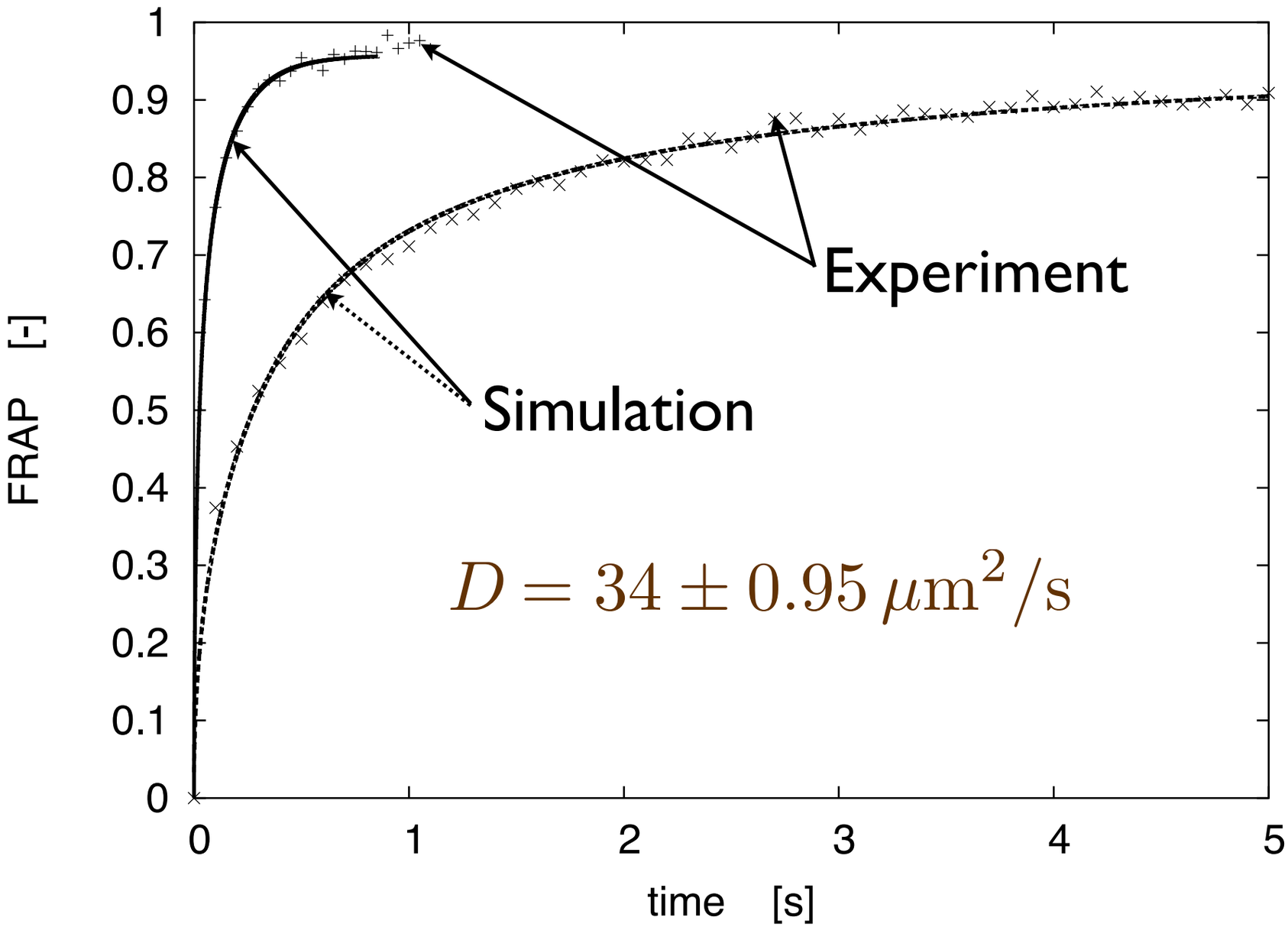






Sbalzarini et al., Biophys. J. 89(3):1482. 2005.

Simulation vs. Experiment

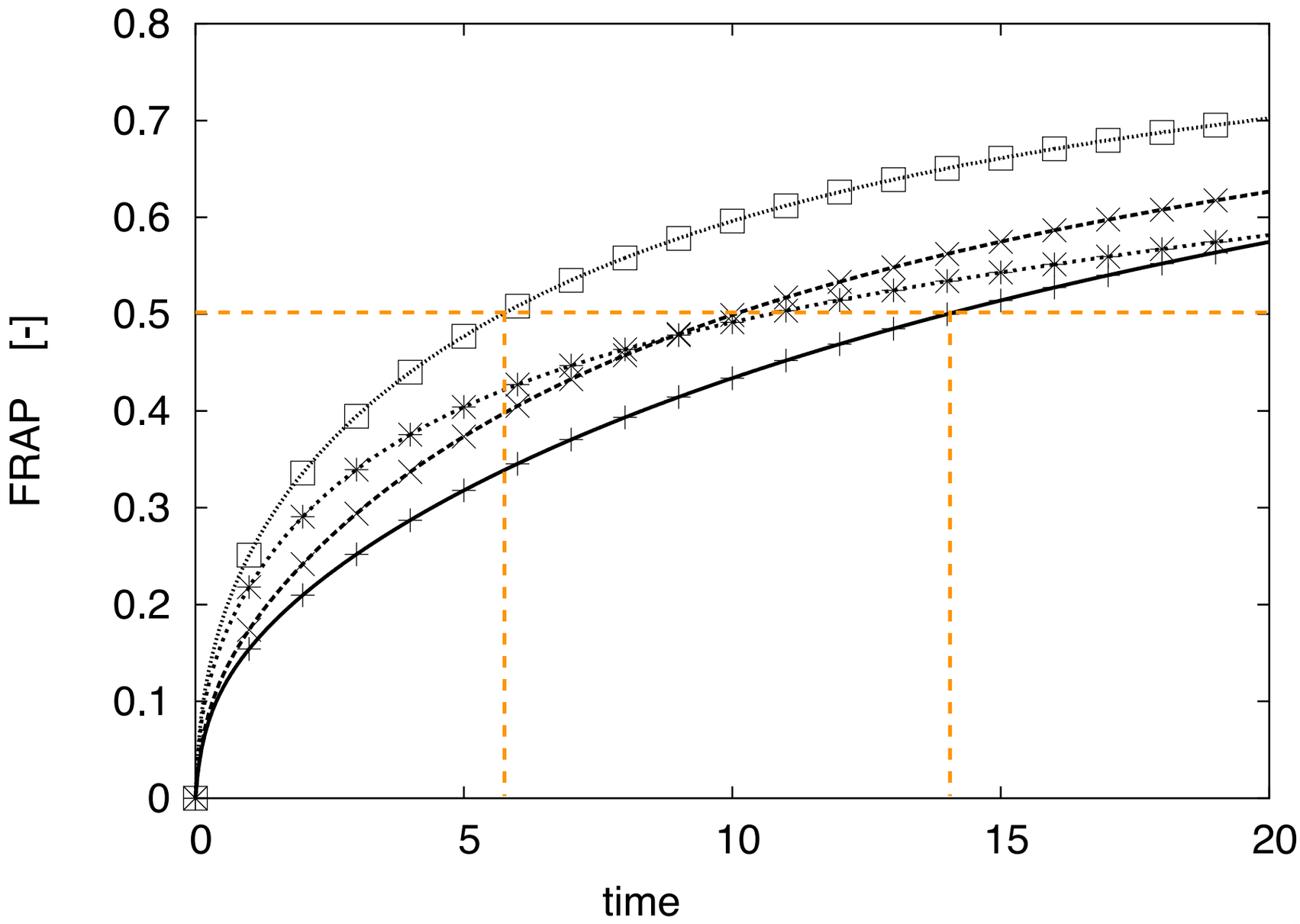


Simulation and Experiment in the **same** ER

***in vivo* diffusion constant from fit**

Sbalzarini et al., Biophys. J. 89(3):1482. 2005.

Beyond Experiments



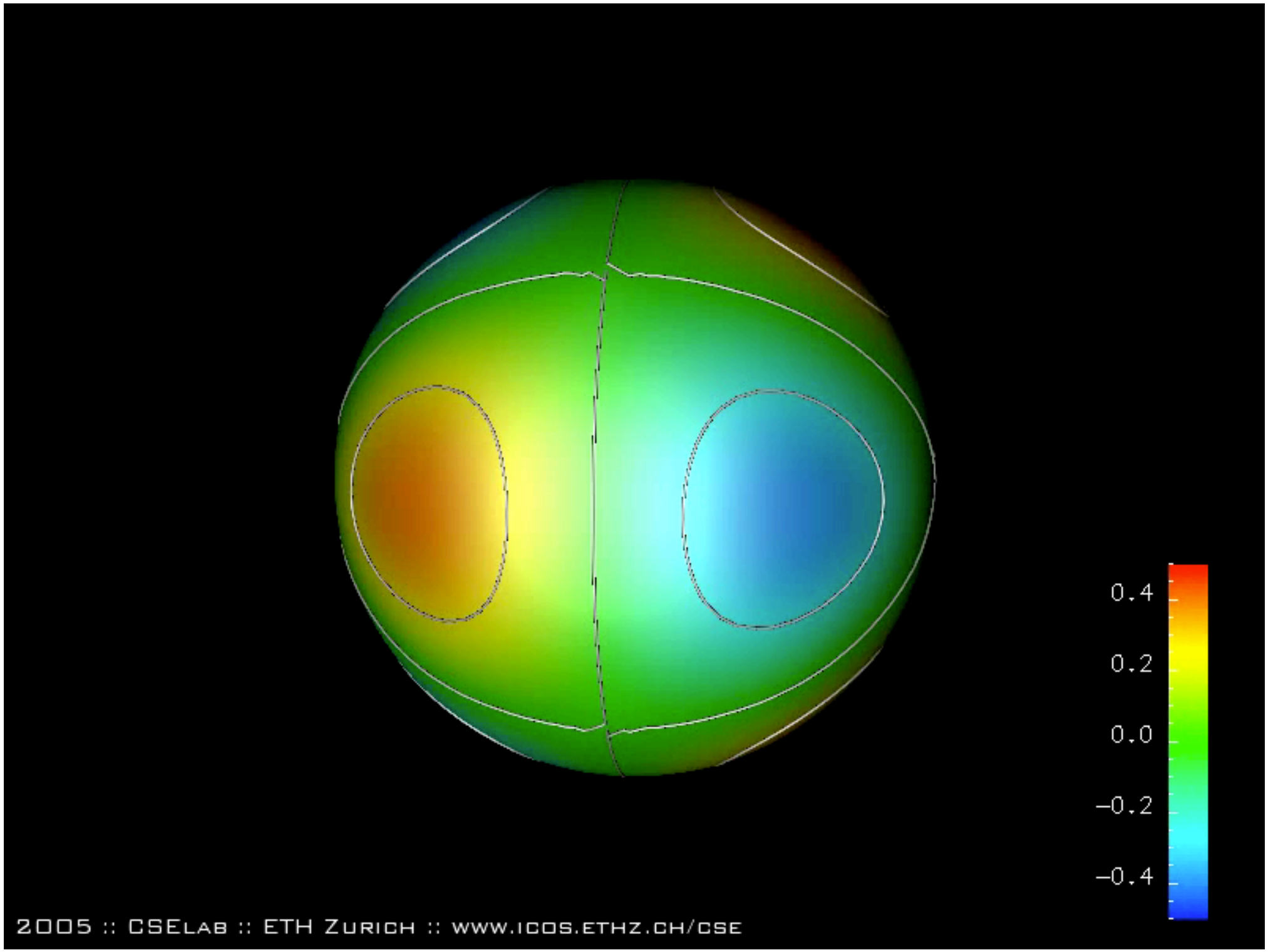
FRAP simulations in the lumen of *different ER*

All simulation use *the same diffusion constant*

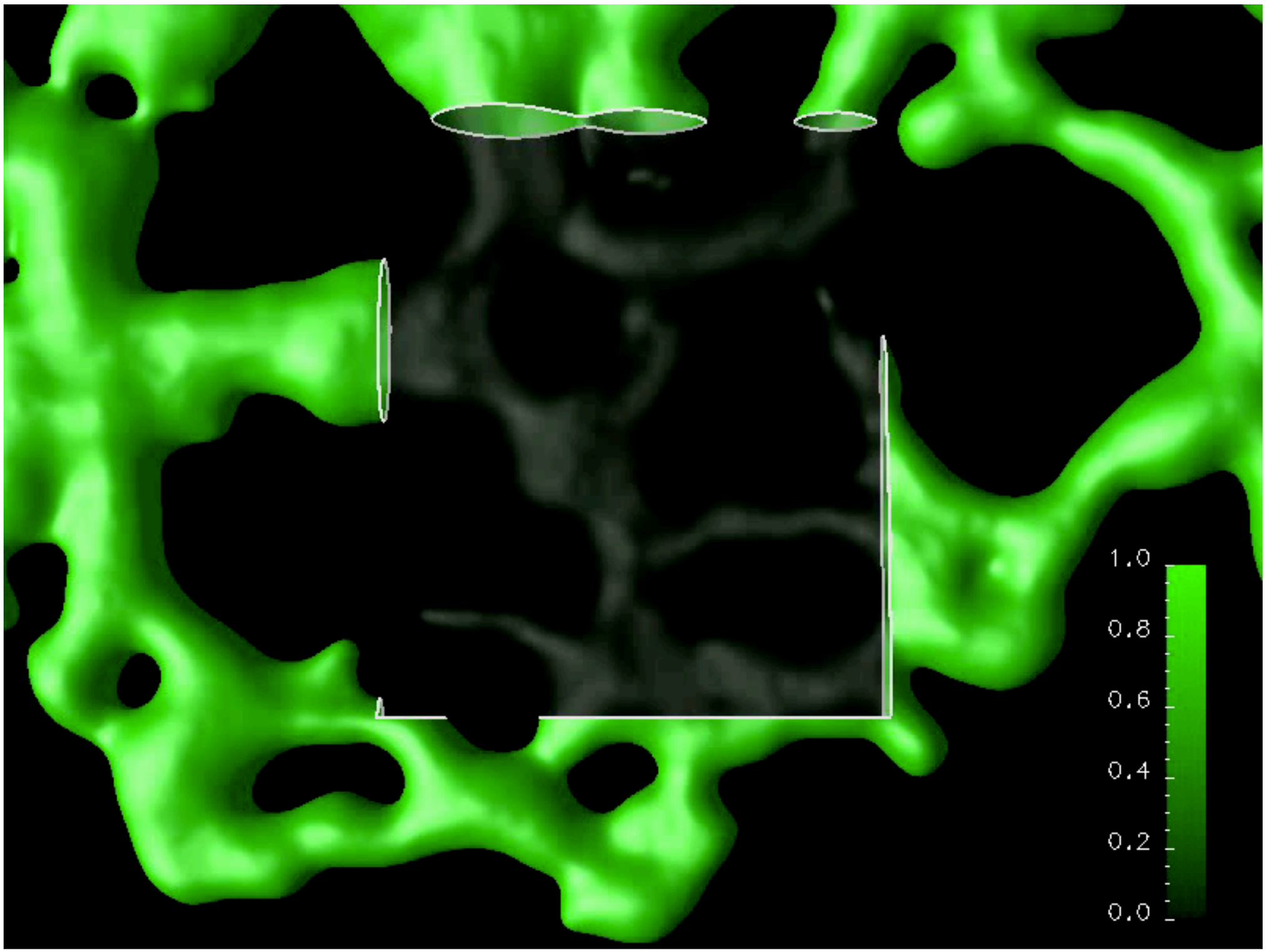
Influence (artifact) of geometry: ***250%***

Sbalzarini et al., Biophys. J. 89(3):1482. 2005.

A new method for solving PDEs on surfaces



Membrane



Sbalzarini et al., Biophys. J. 90(3):1. 2006.

Talk Outline

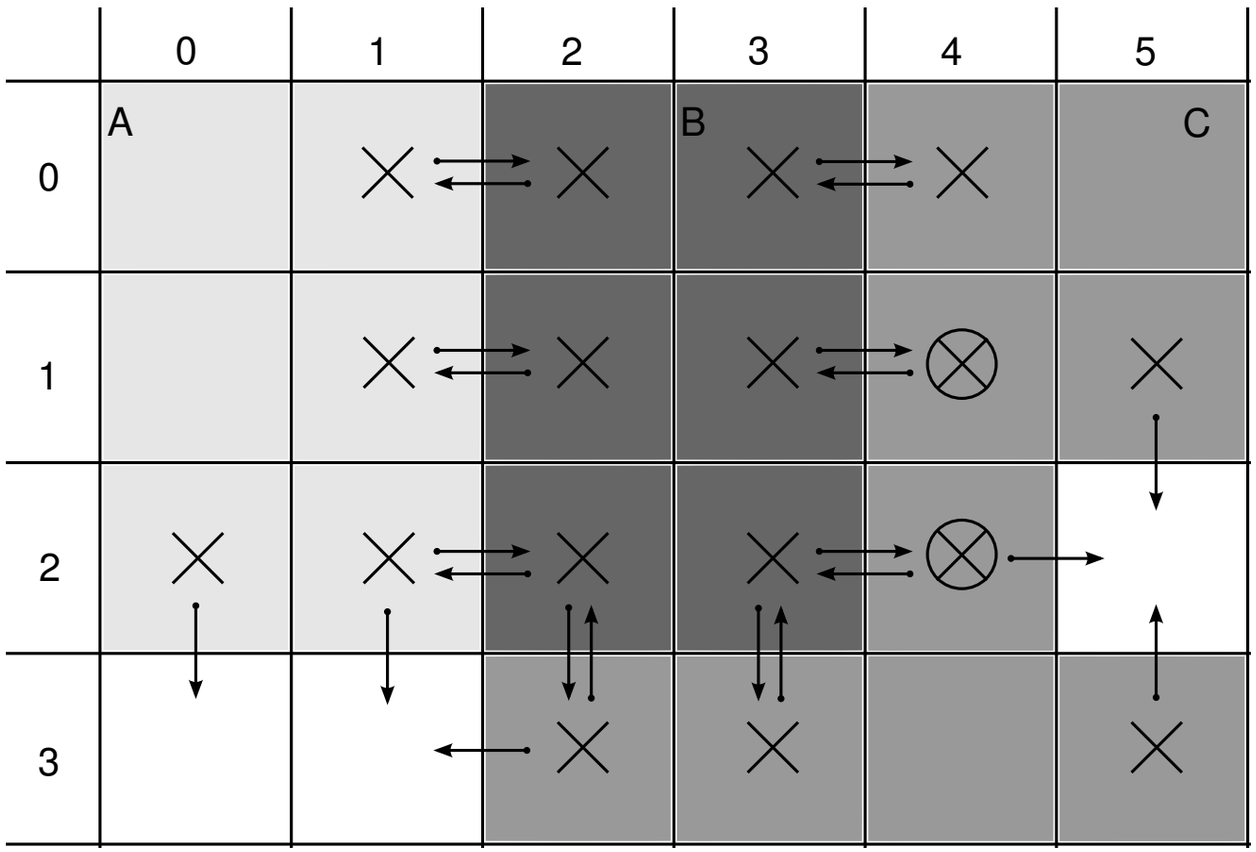
Deterministic particle methods for multi-resolution spatiotemporal simulations

Particle-based bio-image processing

A parallel middleware and domain-specific language for particle-mesh methods on heterogeneous parallel computers

A Particle Method for Image Processing

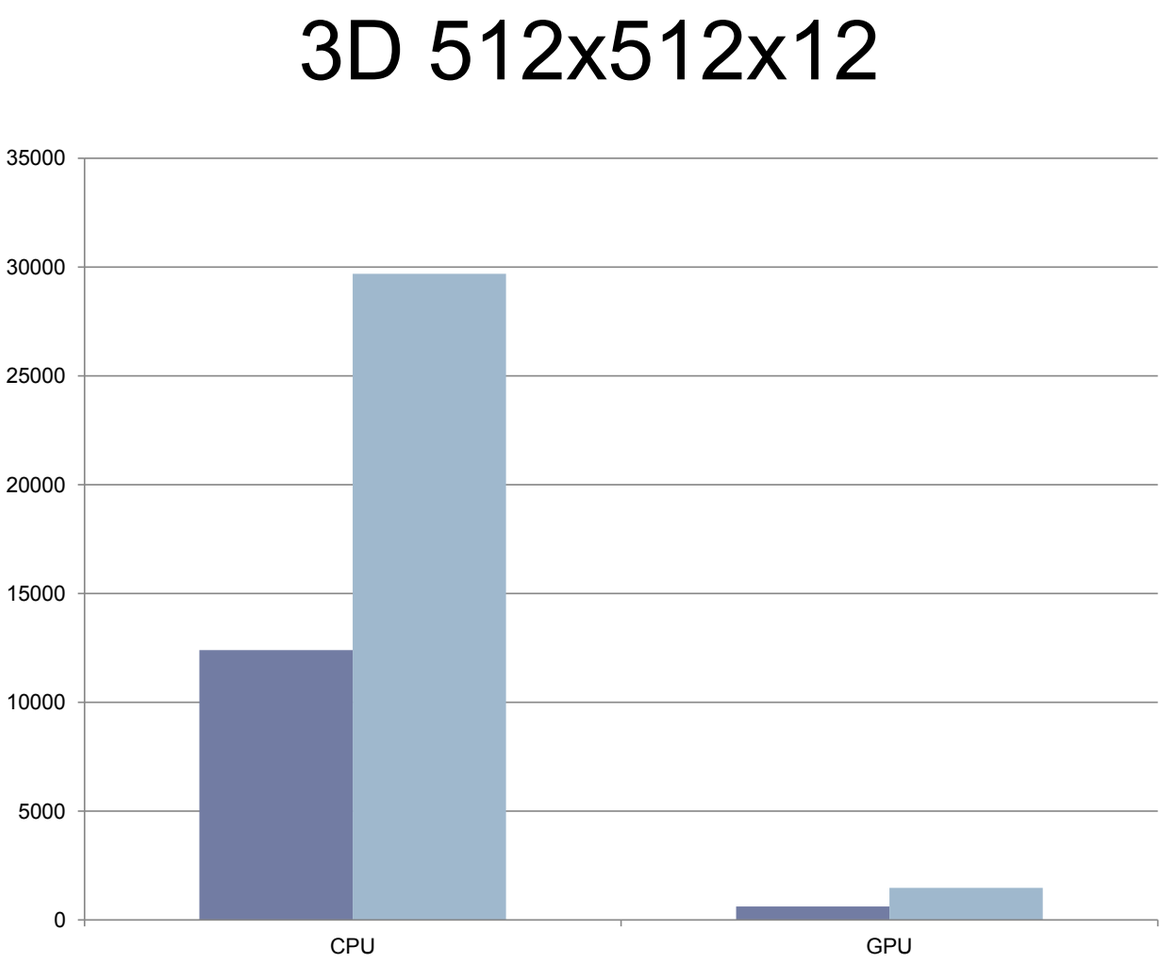
- **Model:** The method is a approximates the gradient and hence optimizes any function.
- **Deformable representation:** Discrete, explicit representation with **labels** (colors) and **particles**:

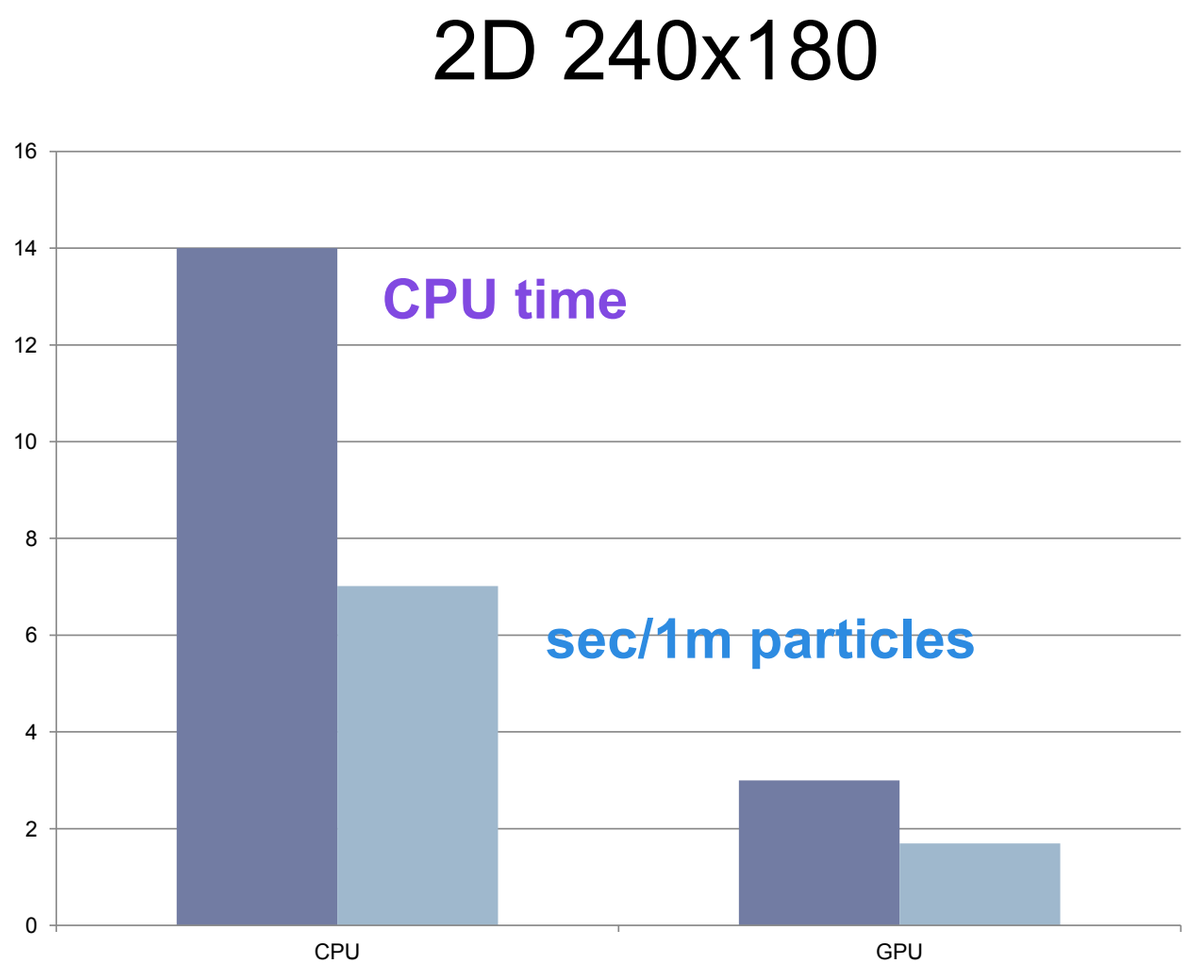


- **Topological control** with *digital topology*

Cardinale, Paul, Sbalzarini, ITIP 2012.

Parallelization over Particles

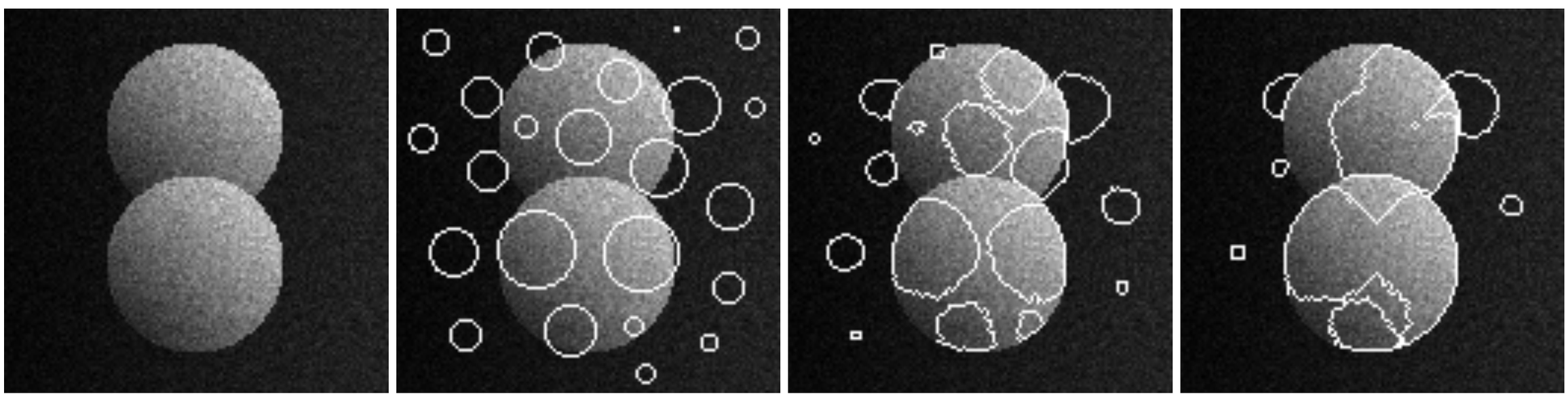




Main source of speedup: use of texture memory!

Eyad Ebrahim, Master Student 2011.

Piecewise Smooth Image Model



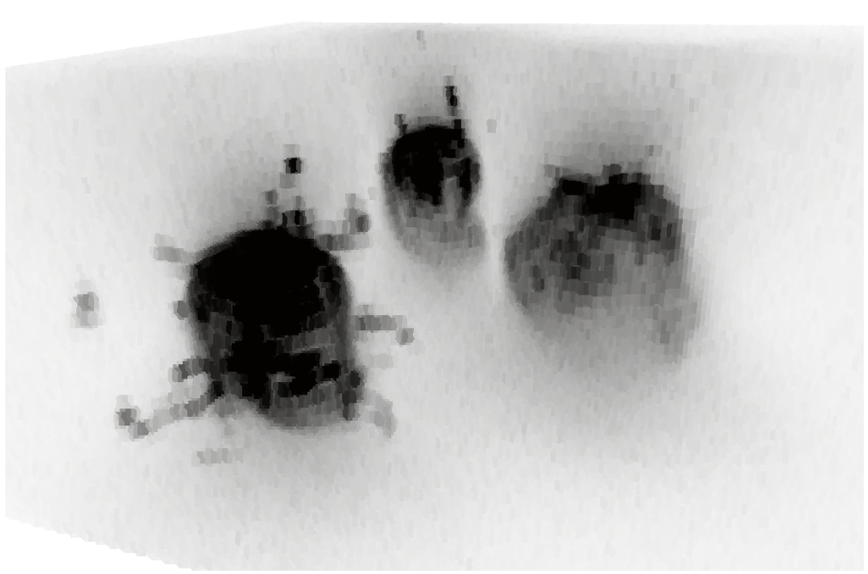
(a) (b) (c) (d)

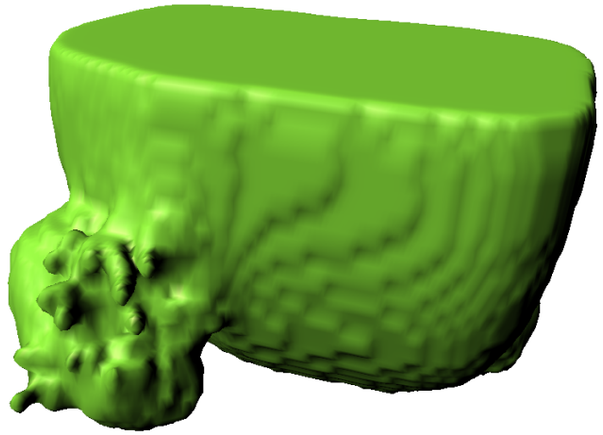
(e) (f) (g) (h)

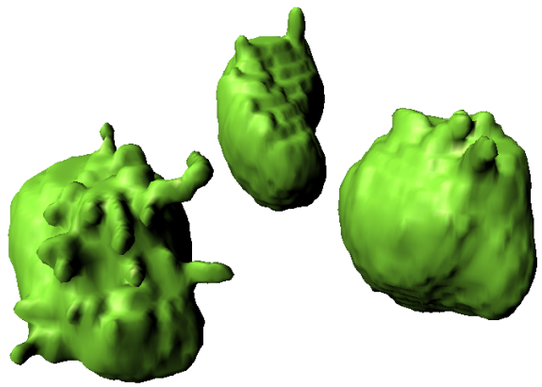
Cardinale, Paul, Sbalzarini, ITIP 2012.

3D Piecewise Smooth Segmentation

- **Splits/Handles** decrease the energy and hence arise naturally during optimization.



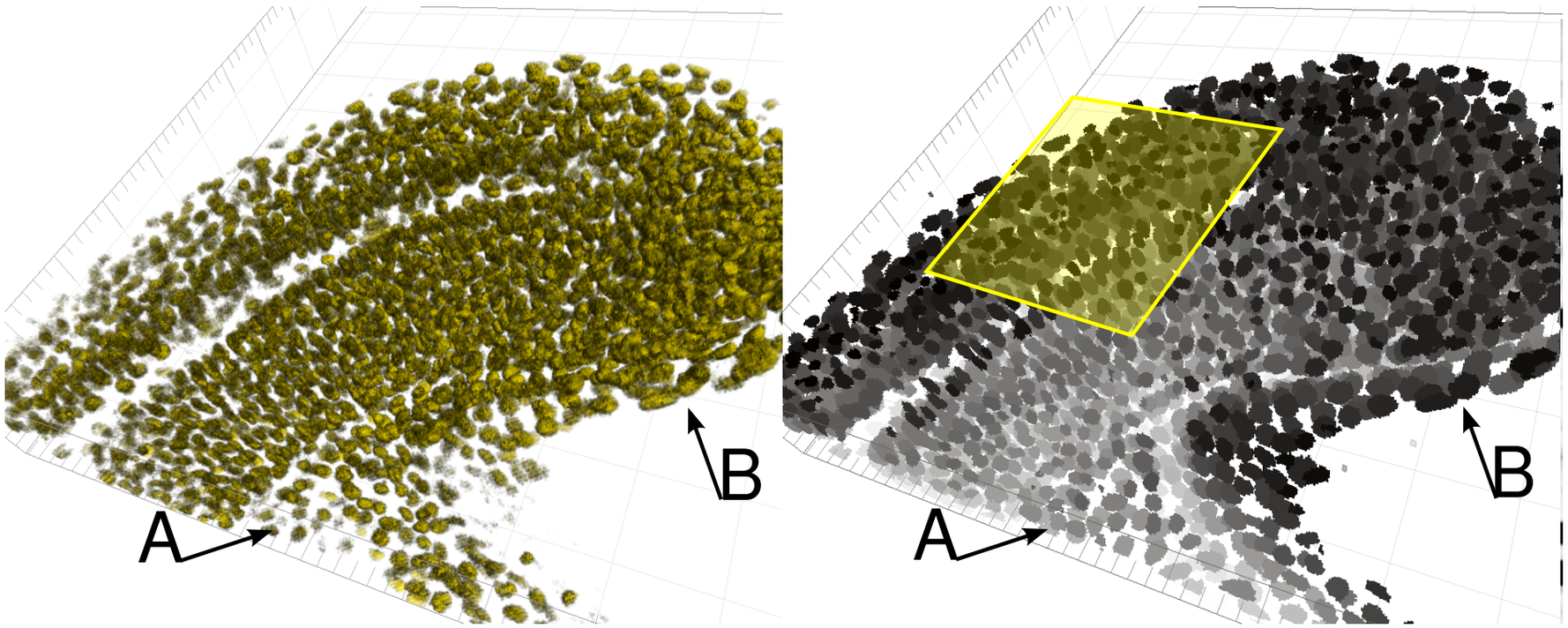


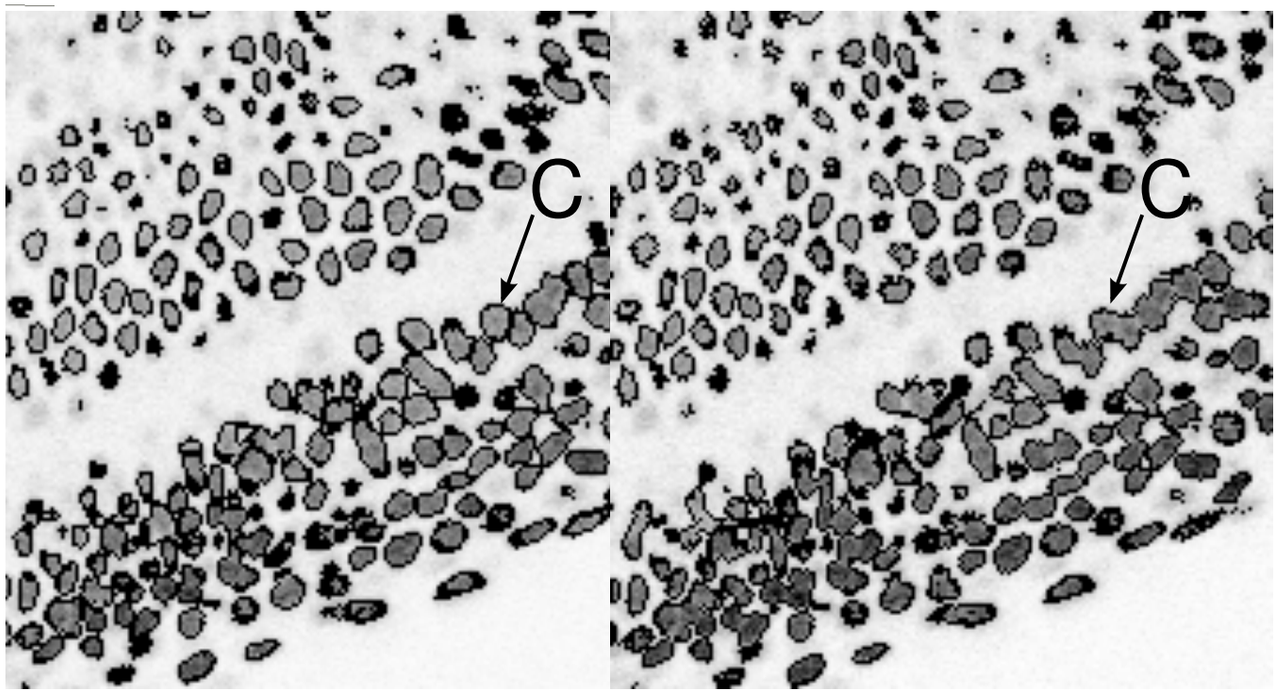


Data: Zebra-fish germ cells. Mohammad Goudarzi, University of Münster

Cardinale, Paul, Sbalzarini, ITIP 2012.

Whole-Embryo Image Analysis





Data: Andy Oates, MPI-CBG

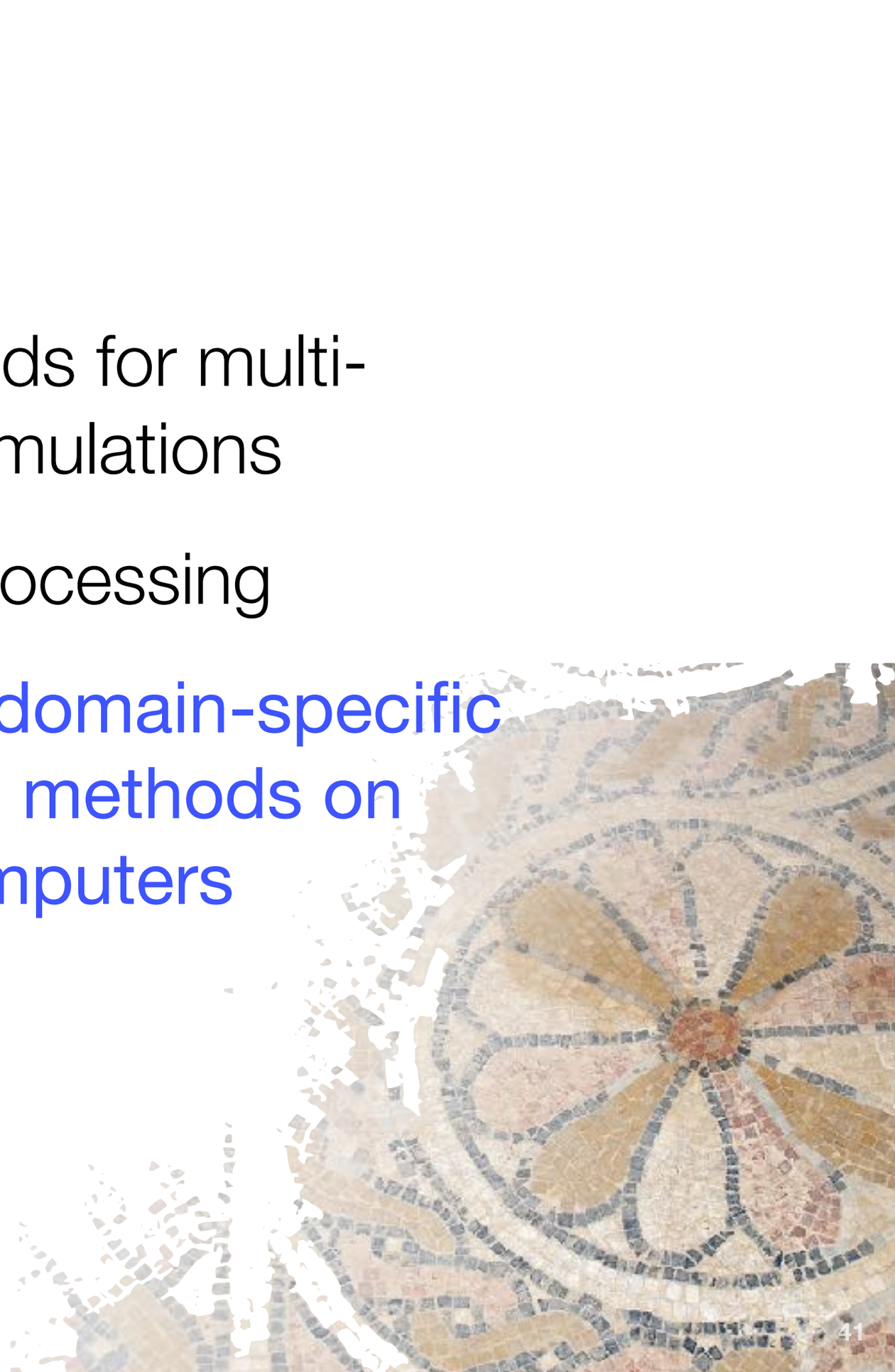
Cardinale, Paul, Sbalzarini, ITIP 2012.

Talk Outline

Deterministic particle methods for multi-resolution spatiotemporal simulations

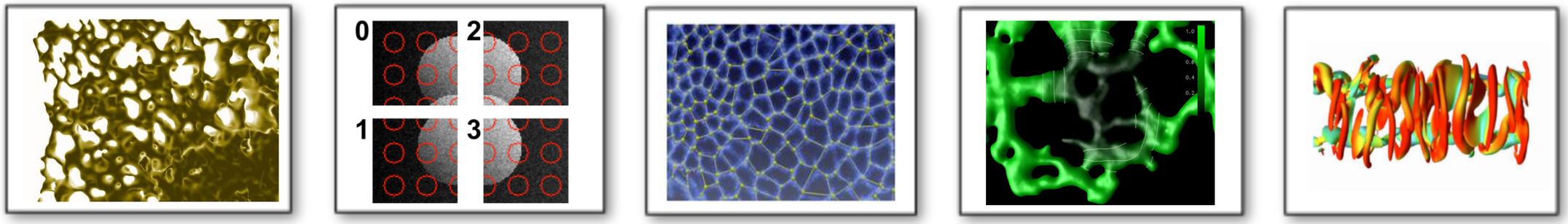
Particle-based bio-image processing

A parallel middleware and domain-specific language for particle-mesh methods on heterogeneous parallel computers



OpenFPM

Middleware implementation of particle-mesh abstractions.



Open Particle Mesh Environment (OpenPME)

DSL optimizer and code generator

OpenFPM core

OpenFPM numerics

C++ Template Metaprogramming

Distributed memory

Shared memory

GPUs

Vector processors

Main Innovation

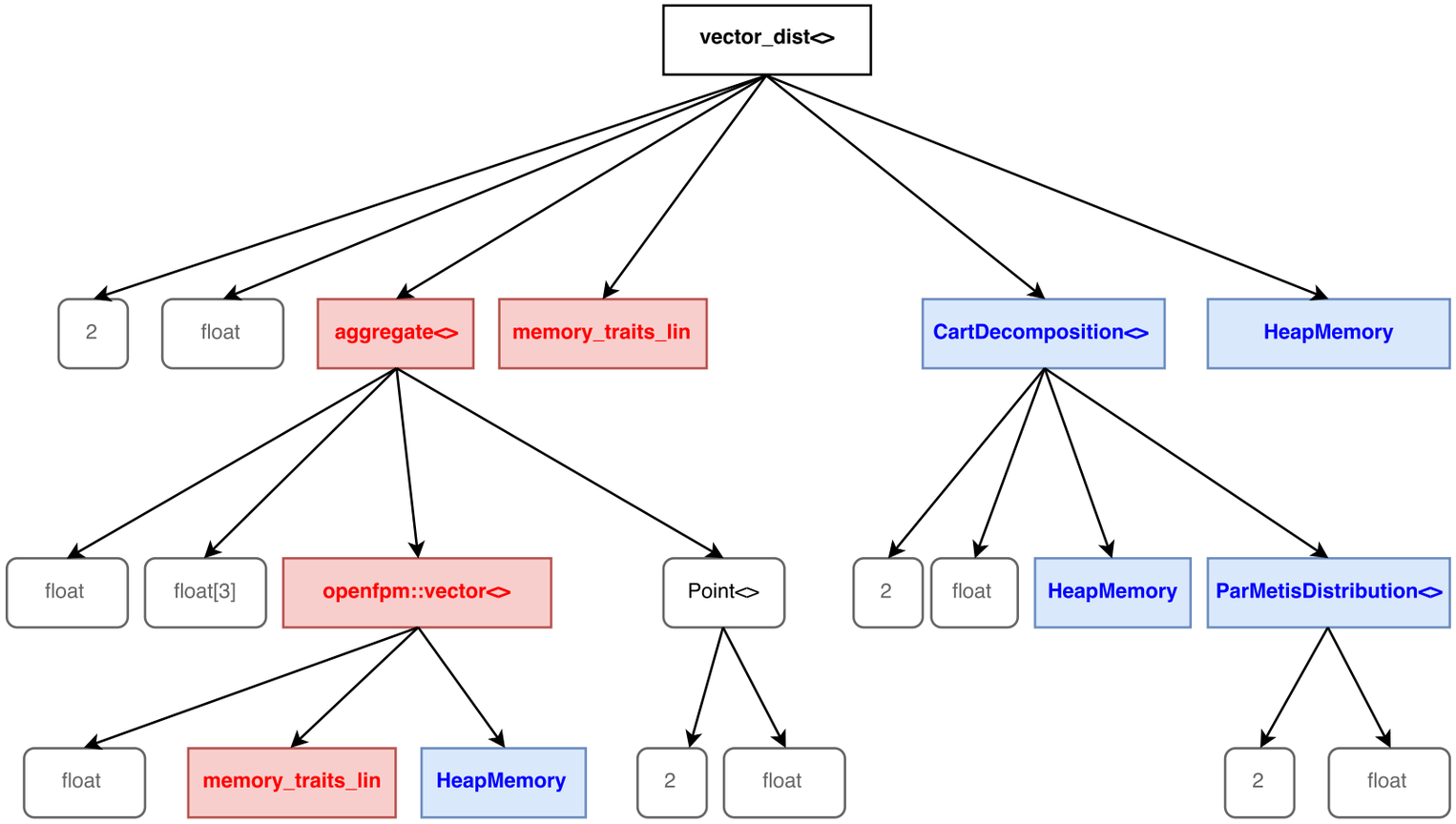
Compile-time code generation

(a)

```
vector_dist<2,float, aggregate<float,
                               float[3],
                               Point<3,double>,
                               openfpm::vector<float>
                              >
           >
```

(b)

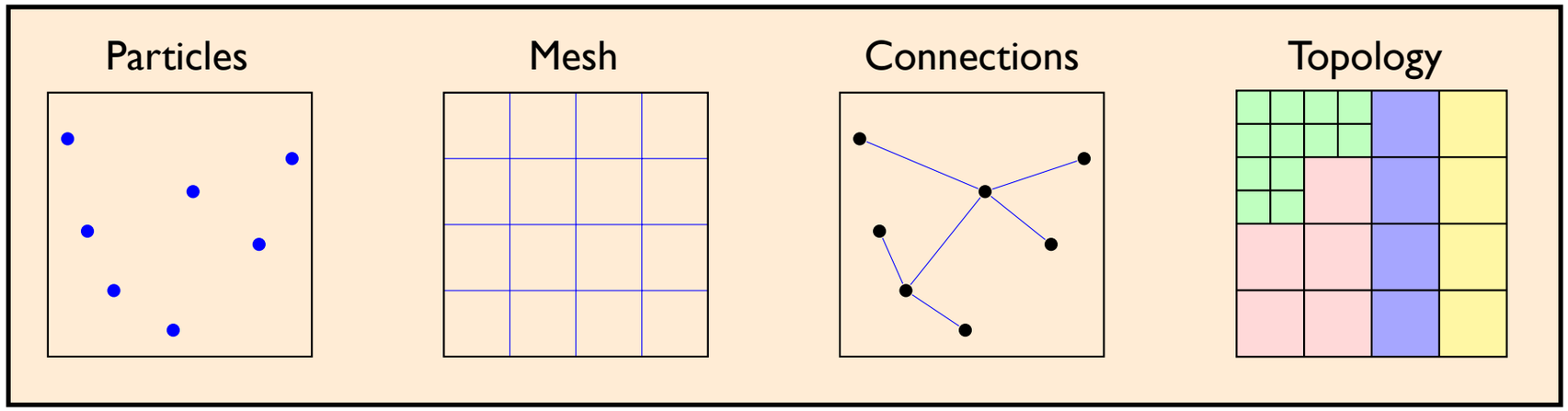
```
vector_dist<2,float, aggregate<float,
                               float[3],
                               Point<3,double>,
                               openfpm::vector<float>>,
            memory_traits_lin,
            CartDecomposition<2,float>,HeamMemory,
                              ParMetisDistribution<2,float>
            HeapMemory
            >
```



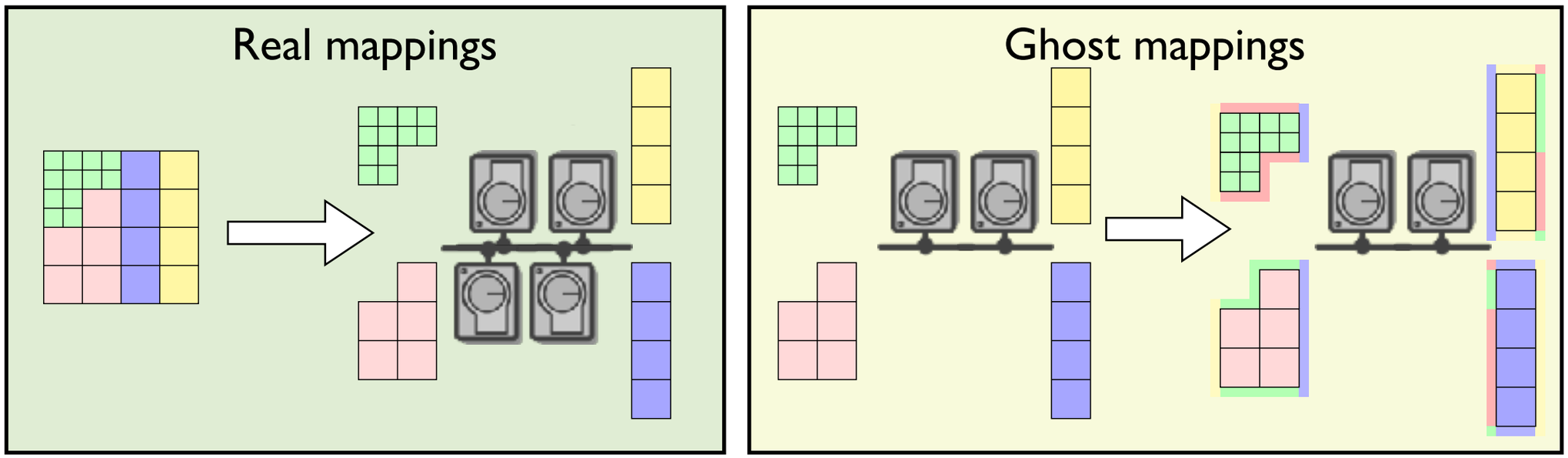
Parallel Particle-Mesh Abstractions

Sbalzarini et al., J. Comput. Phys. 215:556. (2006)
I. F. Sbalzarini, Intl. J. Distr. Systems & Technol. 1(2), 40 (2010)

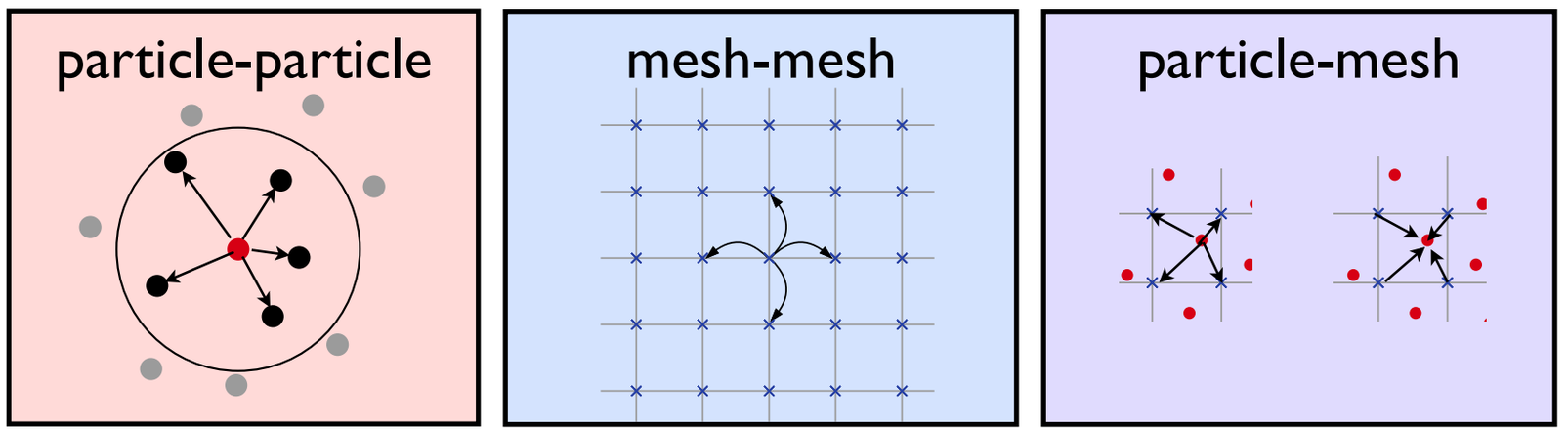
Data abstractions



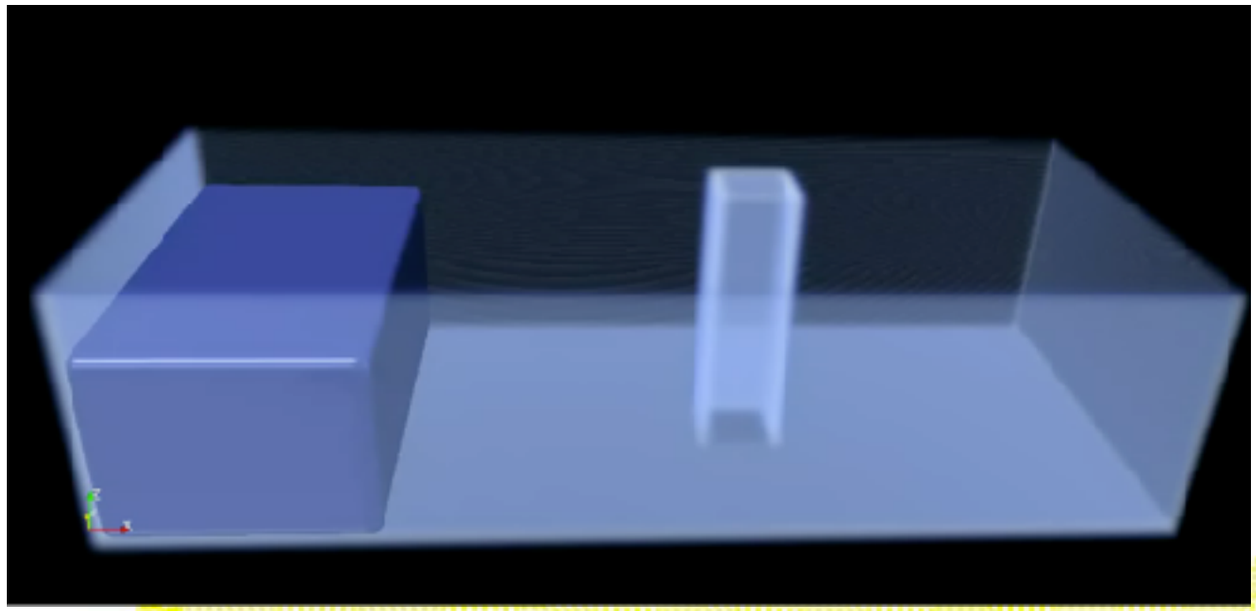
communication abstractions



compute abstractions



Dynamic Load balancing



particles.decompose()

particles.redecompose()

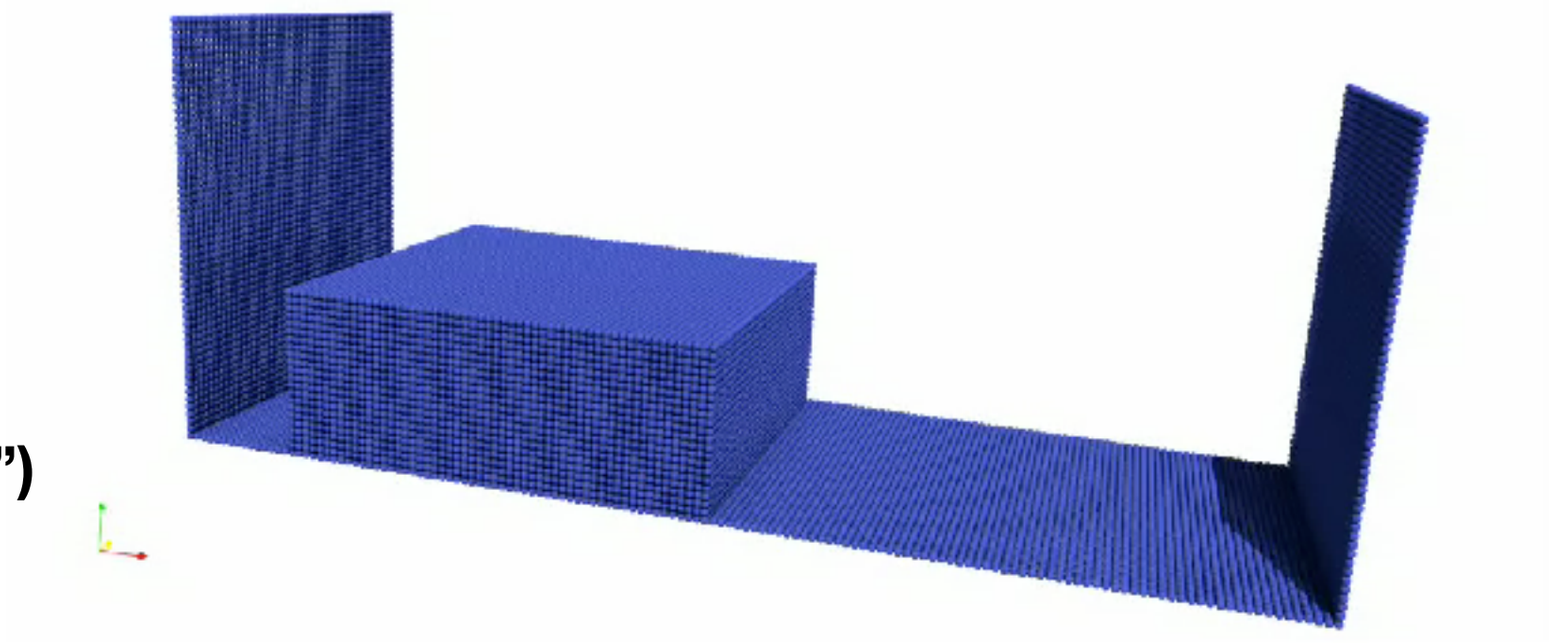
CPU 0
CPU 1
CPU 2
CPU 3

File I/O

VTK: Visualization

grid.write(“my_file”)

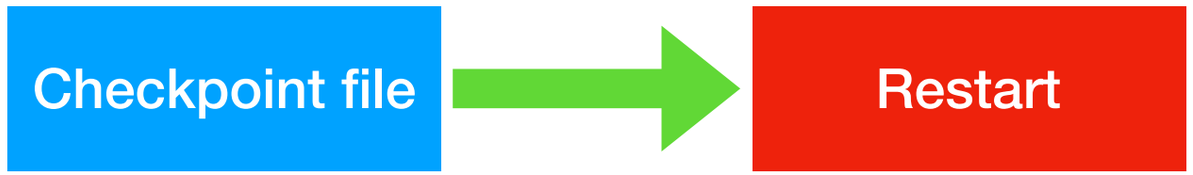
particles.write(“my_file”)



HDF5: Save the full distributed data-structures

grid.save(“my_file”)

particles.load(“my_file”)



Lennard Jones Code Example

Constants

```
double sigma12;
double sigma6;
double r_cut2;

constexpr int velocity_prop = 0;
constexpr int force_prop = 1;
```

Particles definition

```
typedef aggregate<Point<3,double>,Point<3,double>> part_prop;

vector_dist<3,double, part_prop > particles(0,box,bc,ghost);
```

Initialization

```
Ghost<3,float> ghost(r_cut);

 ln_force lennard_jones;
```

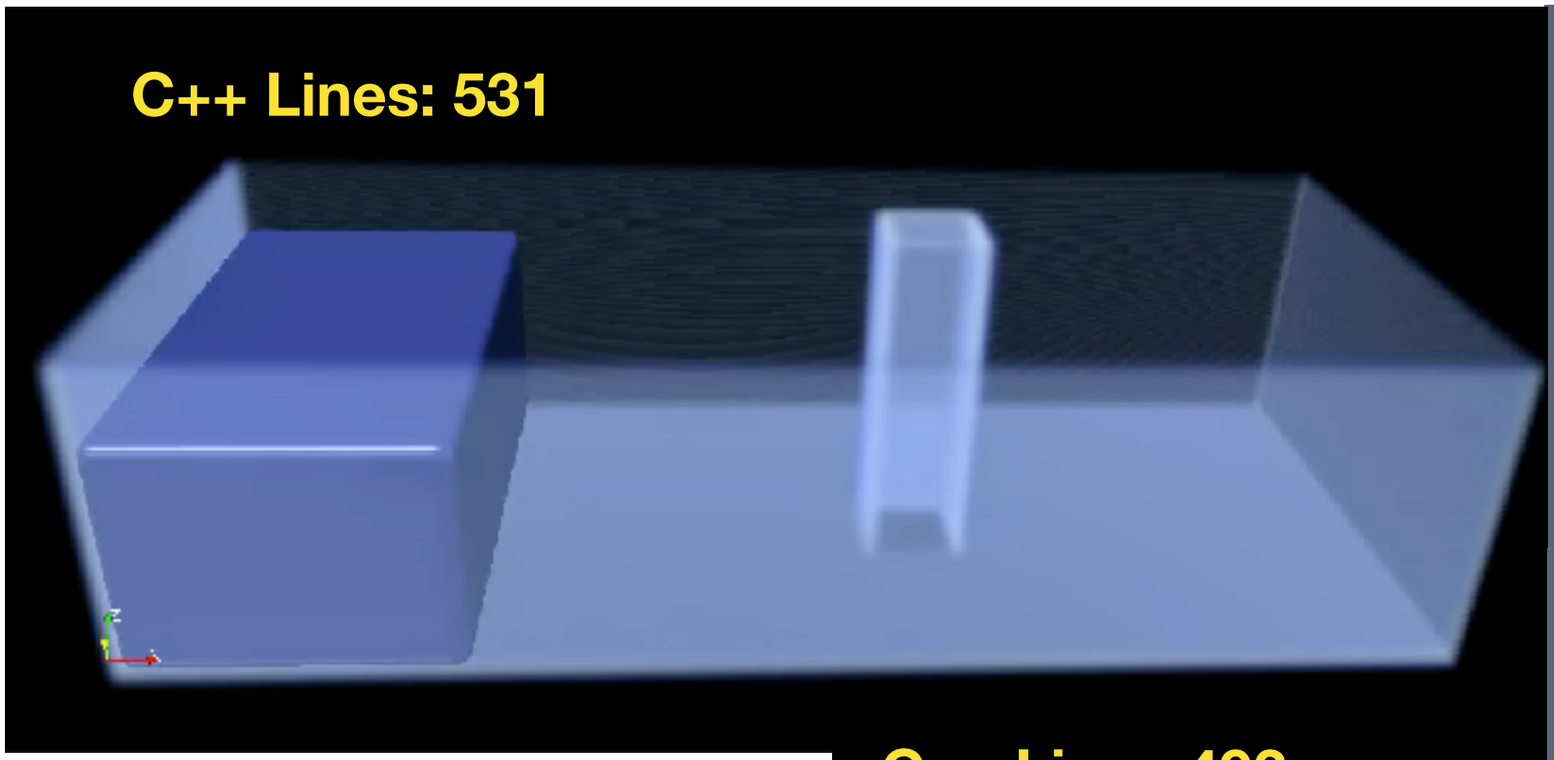
Verlet-list velocity algorithm

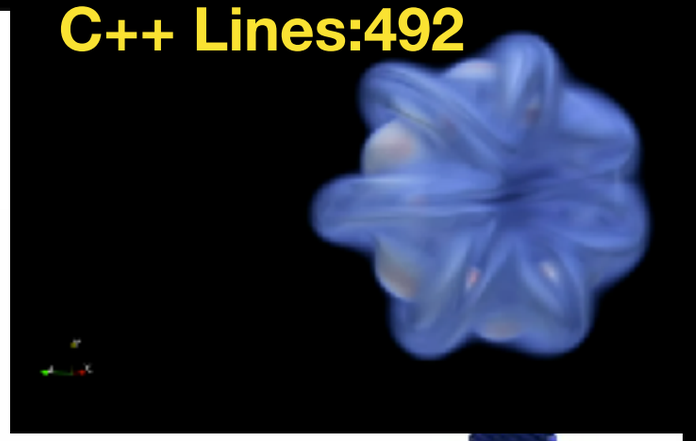
1. Calculate $\vec{v}\left(t+\frac{1}{2}\Delta t\right)=\vec{v}(t)+\frac{1}{2}\vec{a}(t)\,\Delta t\,.$
2. Calculate $\vec{x}(t+\Delta t)=\vec{x}(t)+\vec{v}\left(t+\frac{1}{2}\Delta t\right)\,\Delta t\,.$
3. Derive $\vec{a}(t+\Delta t)$ from the interaction potential using $\vec{x}(t+\Delta t)$.
4. Calculate $\vec{v}(t+\Delta t)=\vec{v}\left(t+\frac{1}{2}\Delta t\right)+\frac{1}{2}\vec{a}(t+\Delta t)\Delta t\,.$

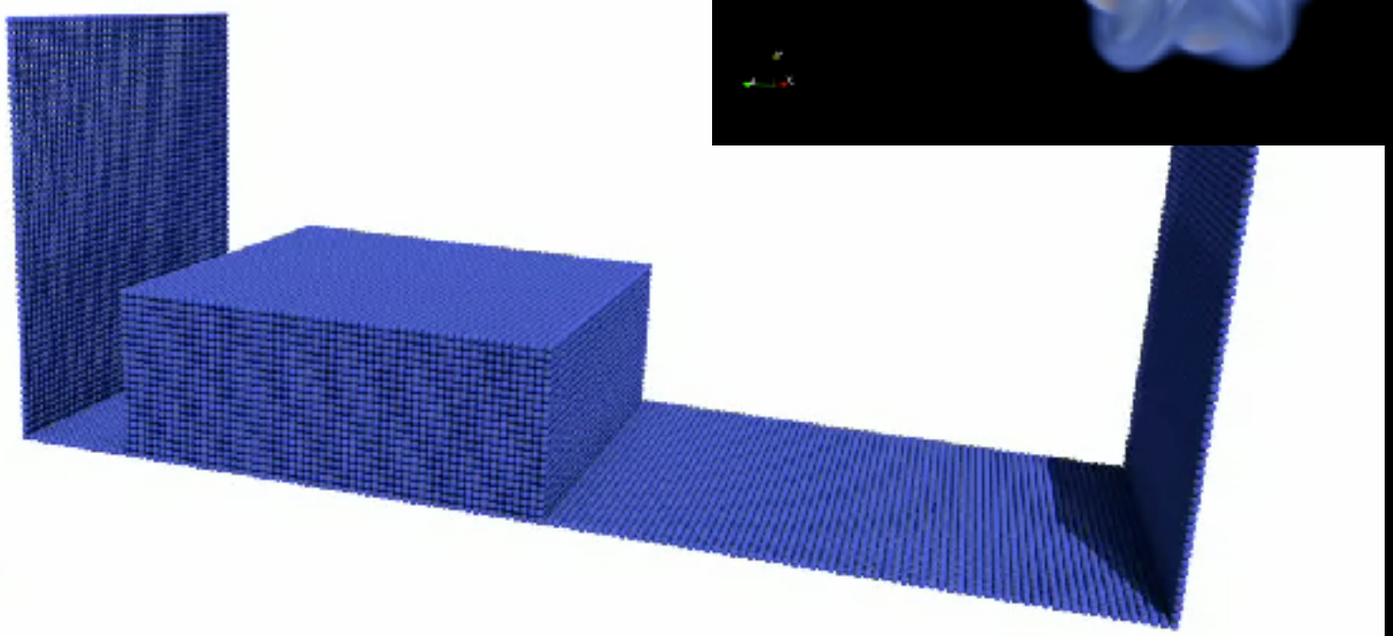
```
for (size_t i = 0; i < 10000 ; i++)
{
   velocity = velocity + 0.5*dt*force;
   position = position + velocity*dt;

   particles.map();
   particles.ghost_get<>();

   particles.updateCellList(NN);
   force = applyKernel_in_sim(particles,NN,lennard_jones);

   velocity = velocity + 0.5*dt*force;
}
```

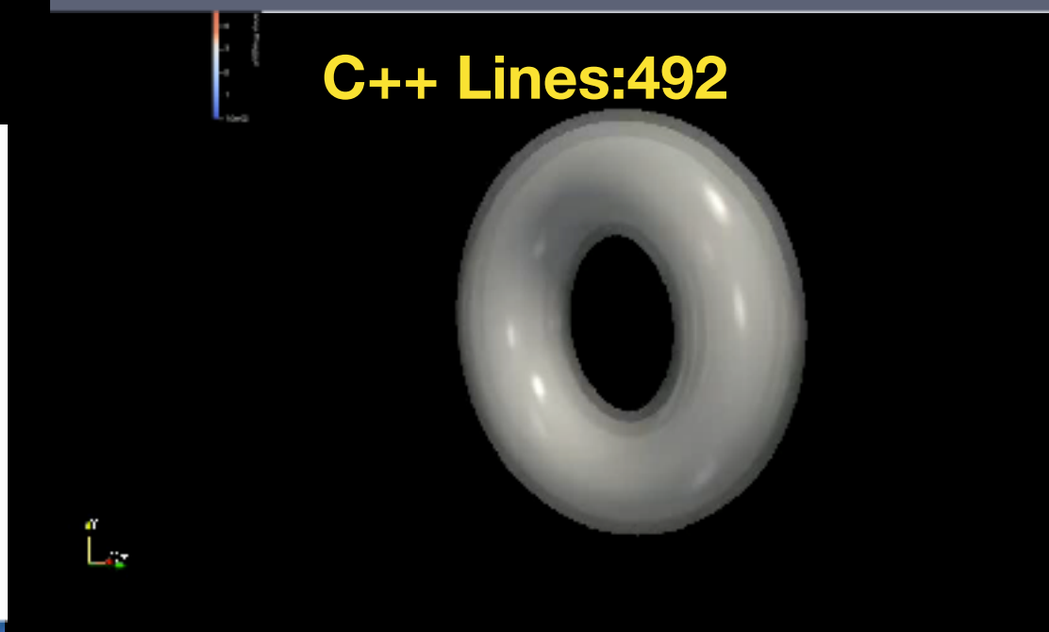
```
  }

  openfpm_finalize();
}
```

Compact scalable simulations

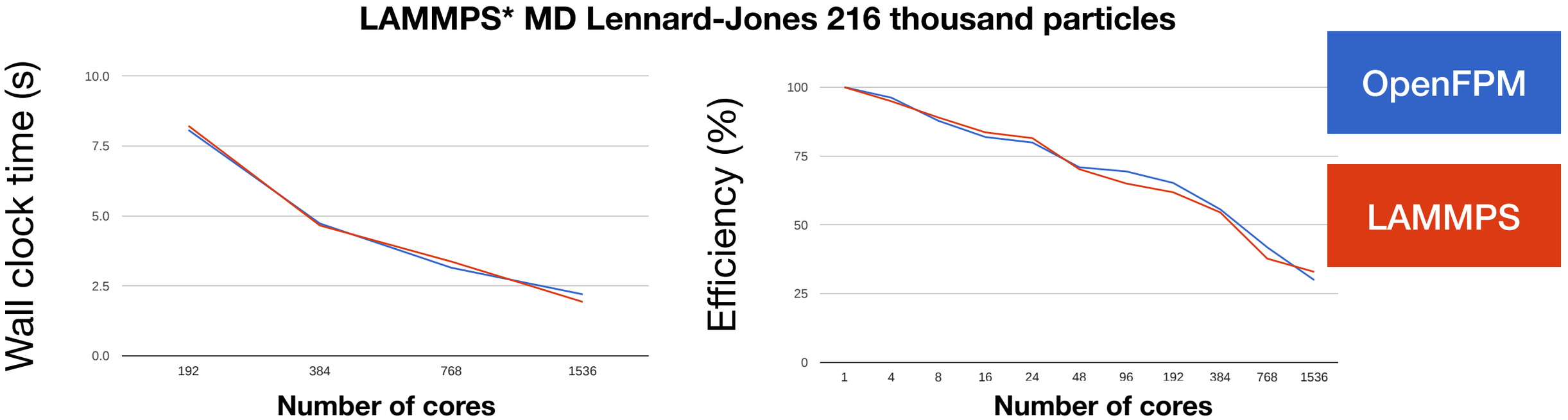


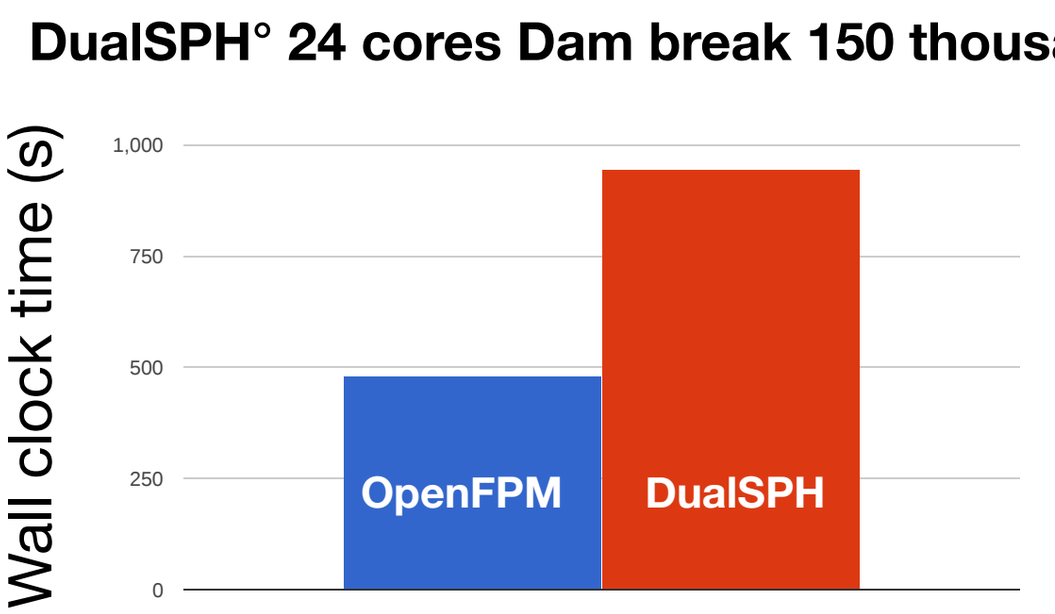


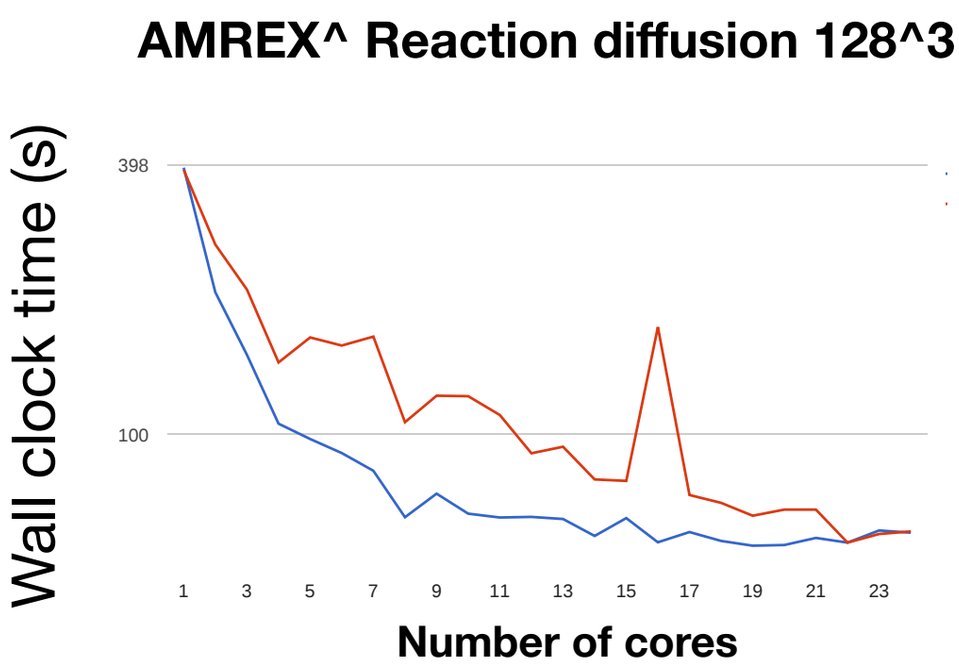




Performance





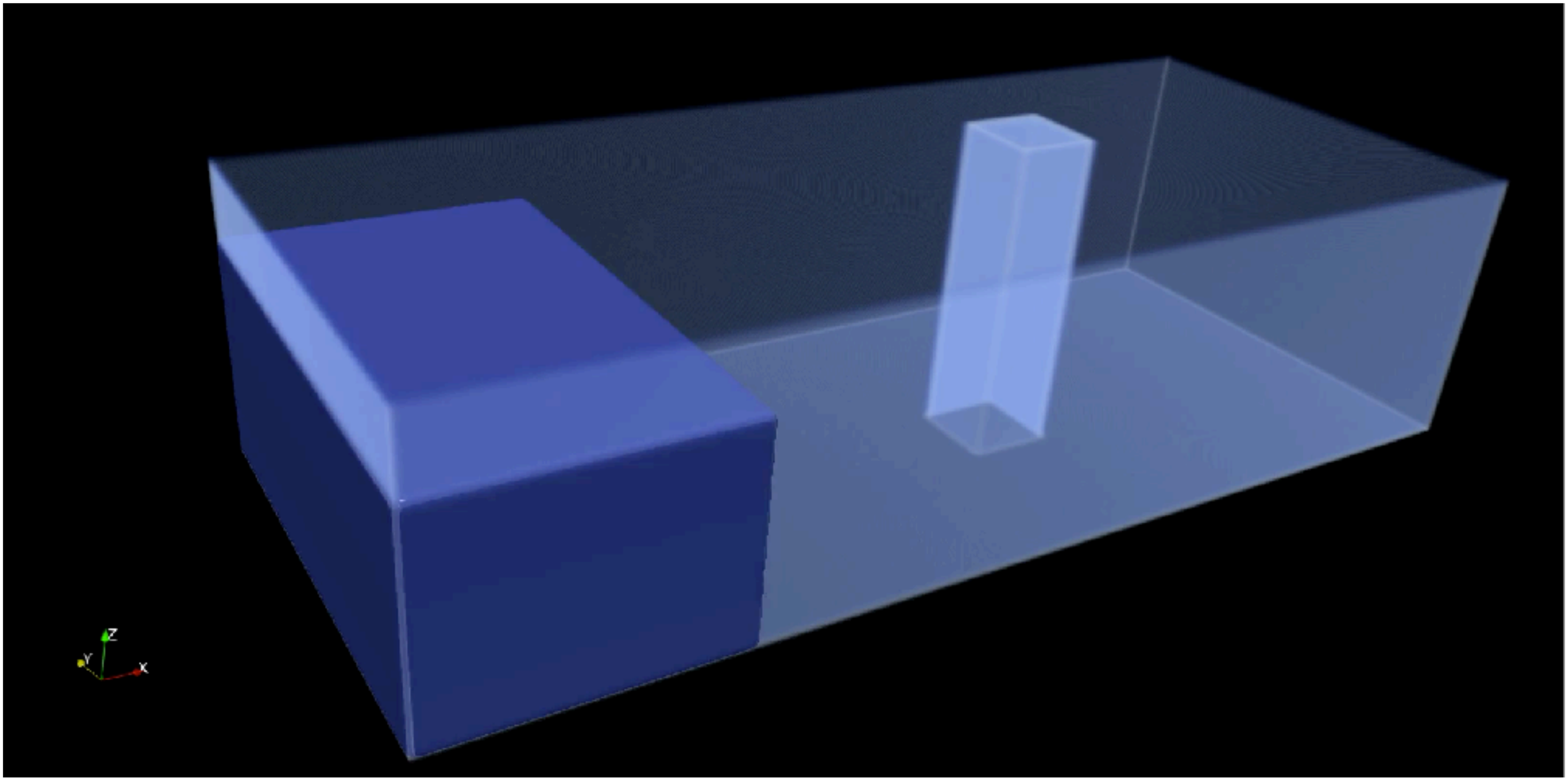


*S. Plimpton, **Fast Parallel Algorithms for Short-Range Molecular Dynamics**, J Comp Phys, 117, 1-19 (1995)

°Crespo et al, **DualSPHysics: open-source parallel CFD solver on Smoothed Particle Hydrodynamics (SPH).** Computer Physics Communications, 2015

^C. A. Rendleman, at al**, Parallelization of structured, hierarchical adaptive mesh refinement algorithms. Computing and Visualization in Science, 3(3):147–157, 2000.**

Multi-GPU automatically



(SPH) GTX: 1080 is 100 to 200 time faster (for float) than E5-2670, 1 core

The PPME Language for parallel computing

Jeronimo Castrillon

ArithmeticOperation × | typeof_BinaryExpression × | typeof_AssignmentExpression × | PartialDerivation × | LaplaceOperator × | GrayScott

```
import LaplacePSE2D from stencils as Lap
import BcdefPeriodic from topologies as bcdef
import RungeKutta from schemes as rk4

module GrayScott
  external real kRate = 1.0;  "reaction rate"
  external real F;            "reaction parameter"
  external real Du = 1.0;     "diffusion constant of U"
  external real Dv = 1.0;     "diffusion constant of V"

  phase initialize
    << ... >>
  end  initialize

  phase solve
    field <real , 1> U;
    field <real , 1> V;
    integer t;

    δU/δt = Du * ∇² U - U *  V²  + F * (1 - U);
    δV/δt = Dv * ∇² V + U *  V²  - (F + kRate) * V;
  end  solve

  phase finalize
    << ... >>
  end  finalize

end  module GrayScott
```

$$\frac{\partial U}{\partial t} = D_u \nabla^2 U - UV^2 + F(1-U)$$

$$\frac{\partial V}{\partial t} = D_v \nabla^2 V + UV^2 + (F+k)V$$

Performance

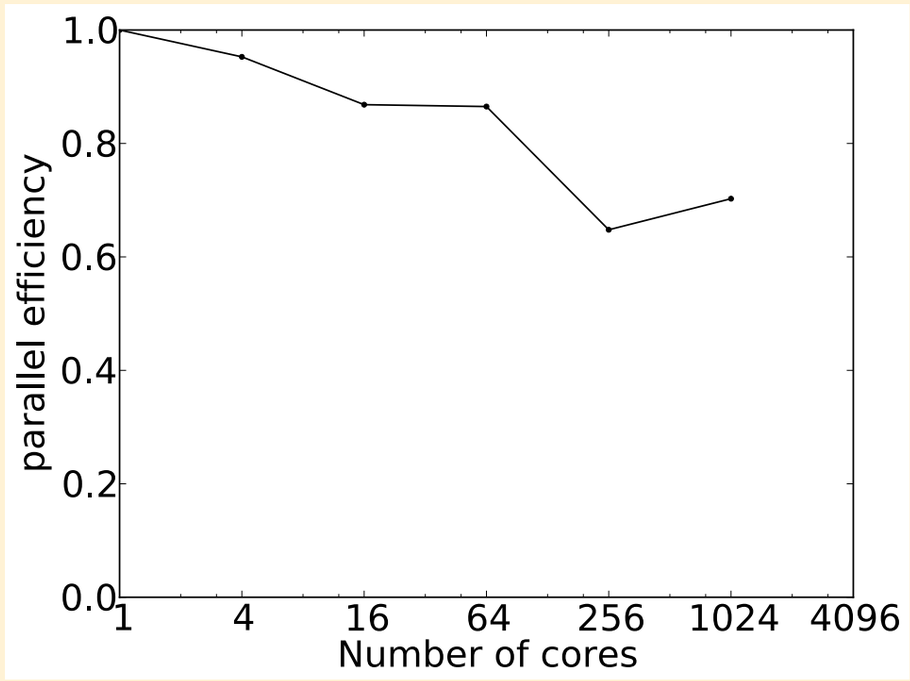
$$\frac{\partial U}{\partial t} =$$
$$\frac{\partial V}{\partial t} =$$

66 F

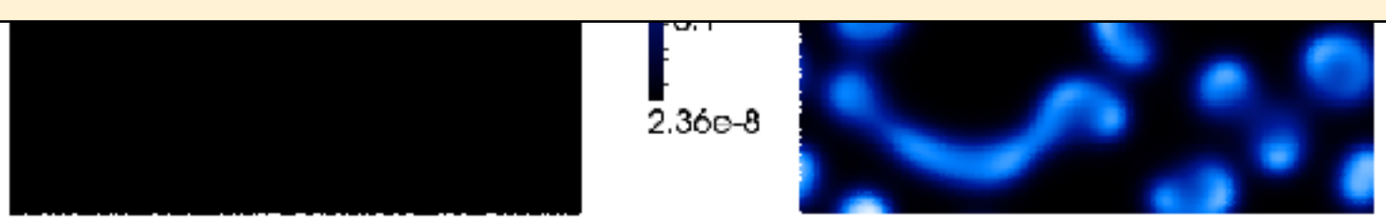
ode

J, LV)

Parallel efficiency



0.21s wall clock time per iteration with 10^6 particles

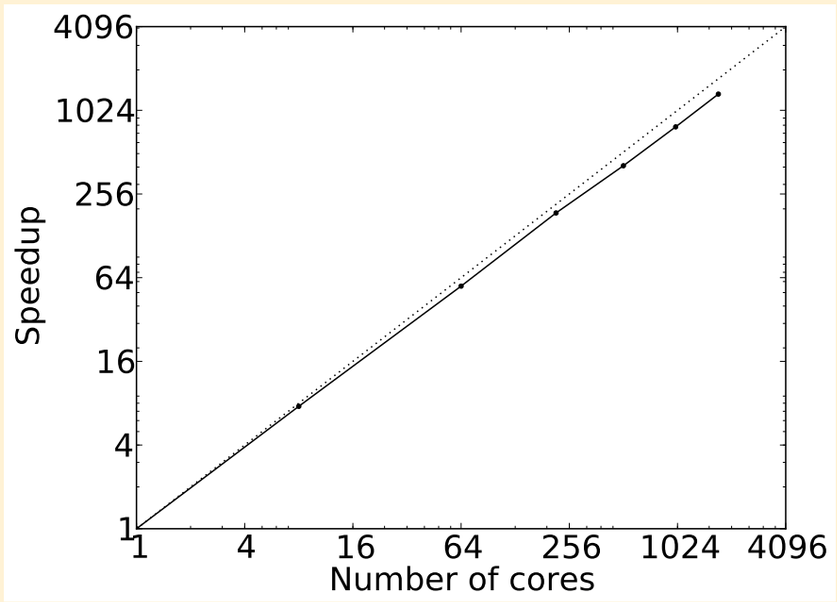


Simulation with 10^5 particles. Turing patterns after 40,000 time steps.

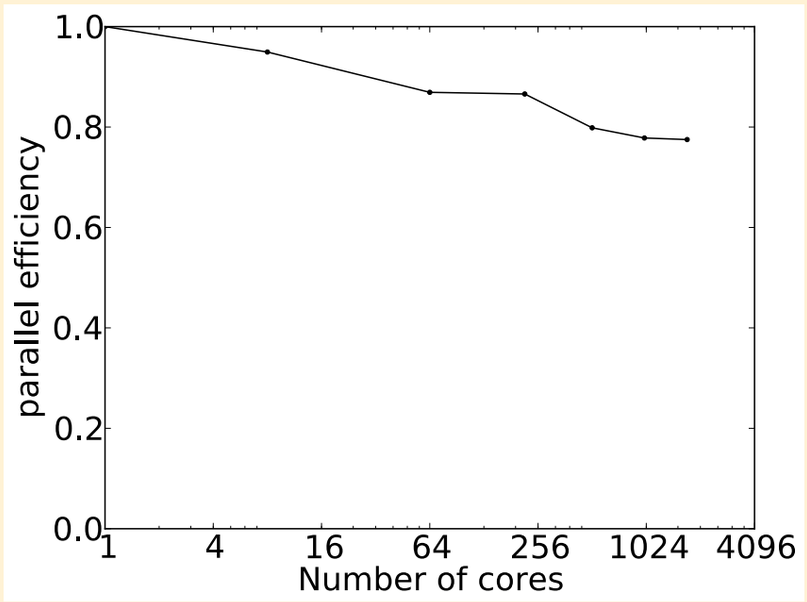
Molecular Dynamics with PPML

The Lennard-Jones potential

Speedup and efficiency with 8 processes per node



speedup



parallel efficiency

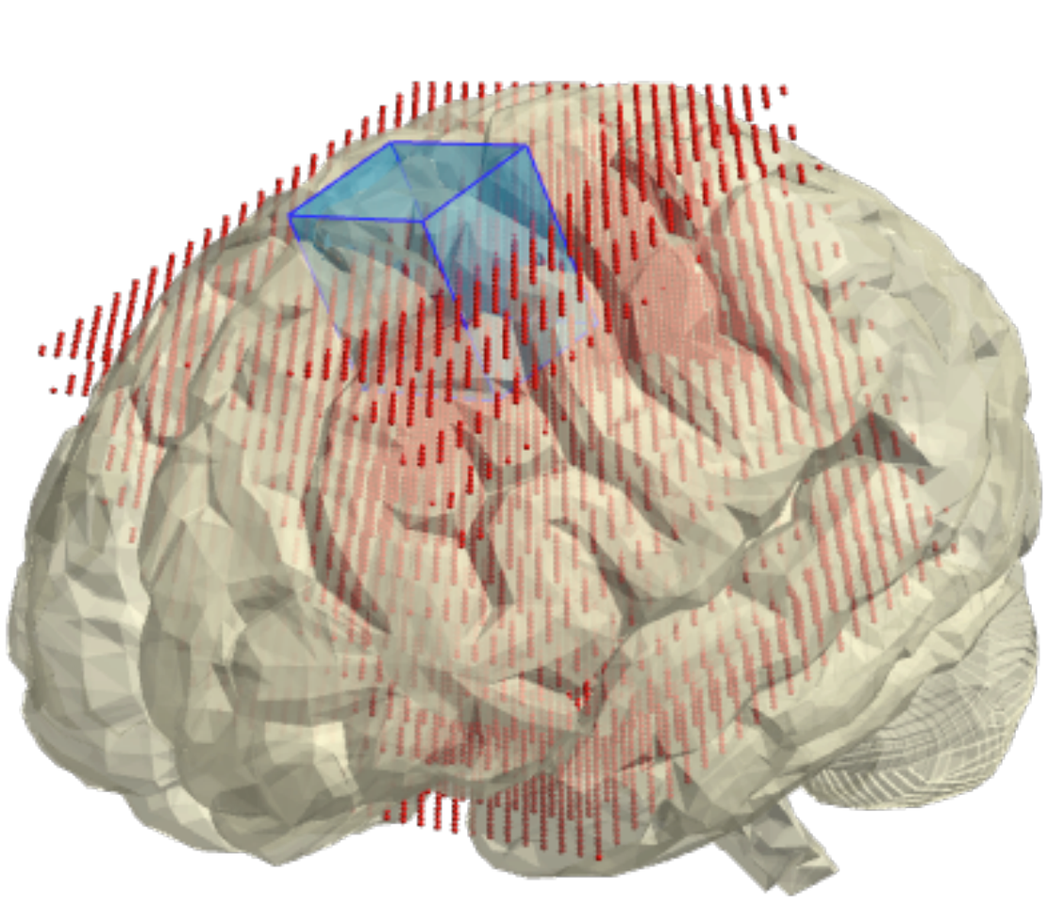
1.43s wall clock time per iteration with 10^6 particles

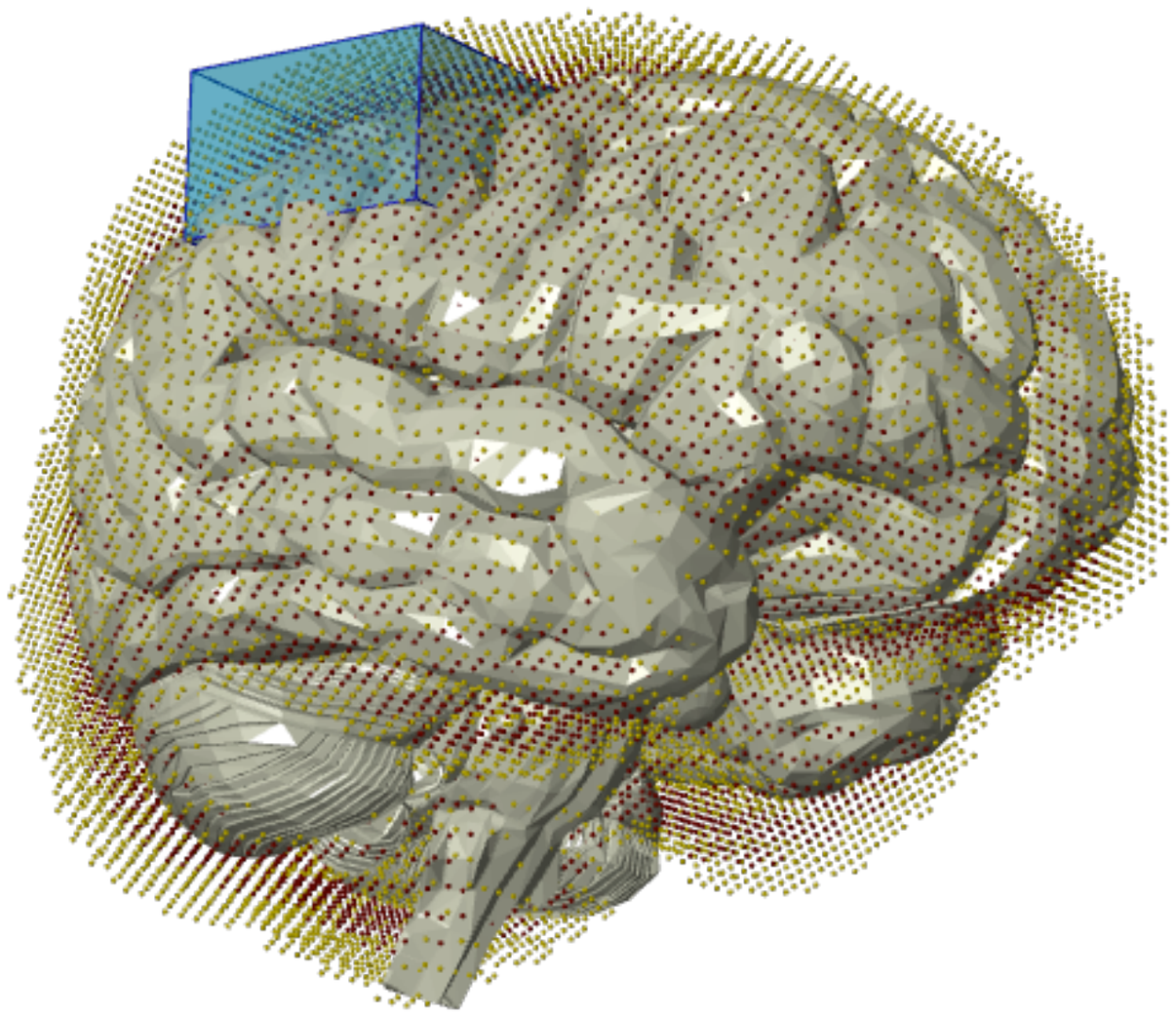
```
ghost_mapping(parts)

nlist = create_neighlist(parts,cutoff=<#cutoff#>,skin=<#skin#>)
```

126 PPML vs. 516 Fortran lines for the code

Brain electromagnetism

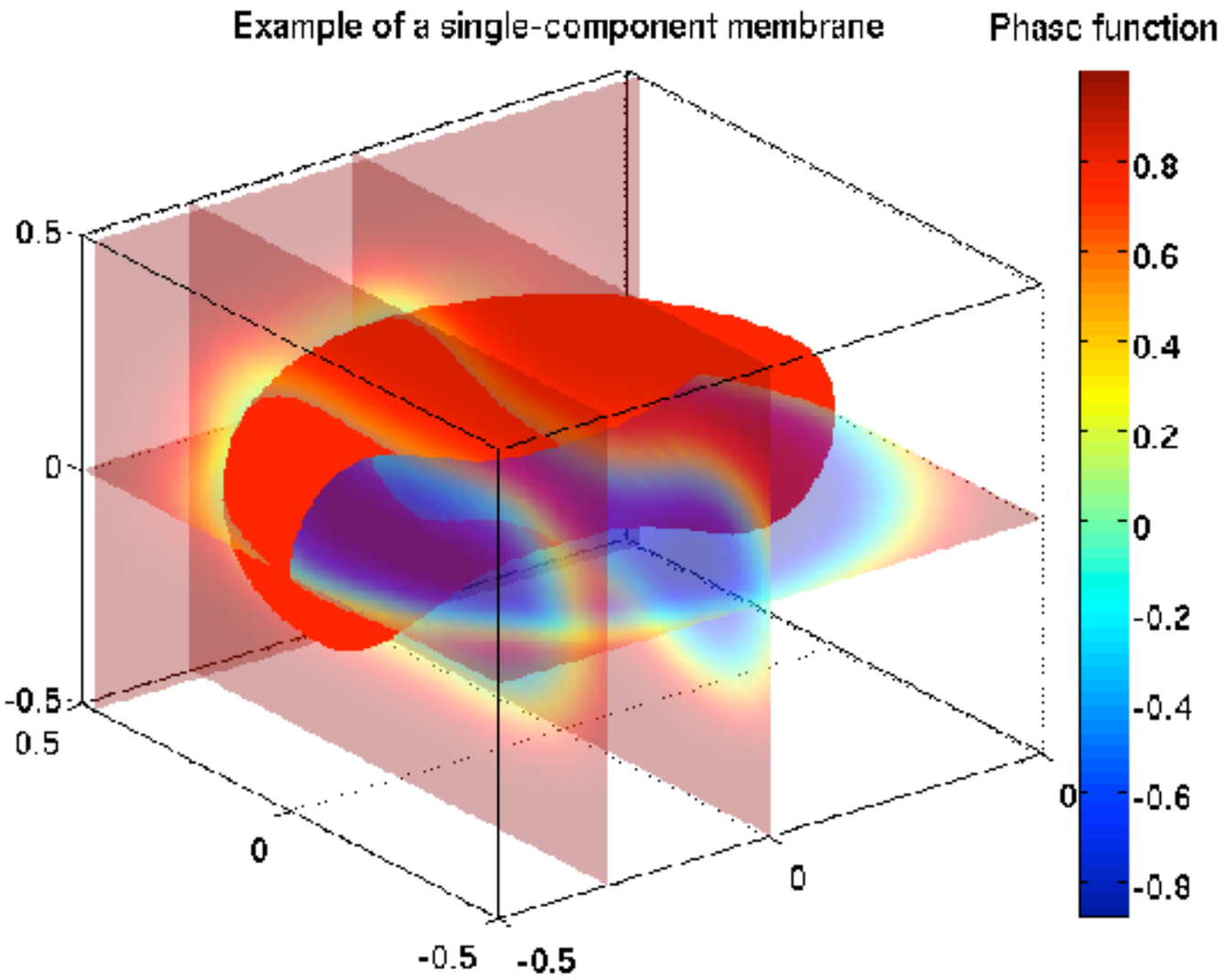
Solve the quasi-stationary Maxwell equations on particle distributions adapted to head/brain geometries from CT/MRI scans





Solve the forward problem for EEG/MEG source localization.

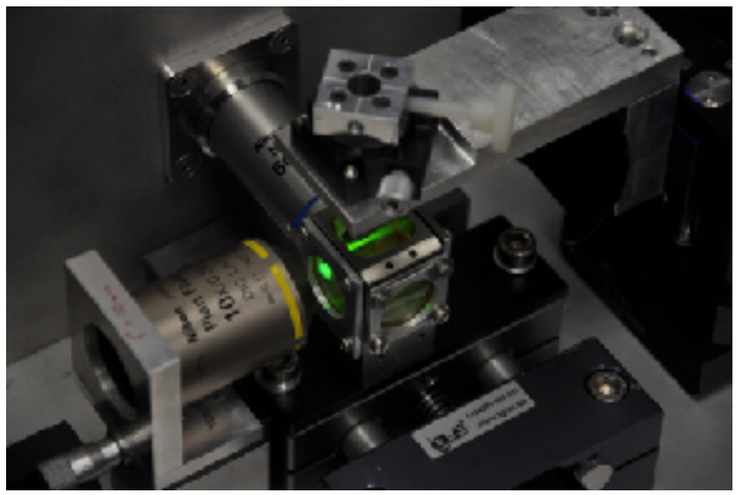
Biomechanics



Equilibrium shape of an Erythrocyte

Simulating the biomechanics of lipid membranes using particle phase field methods

Imagine…

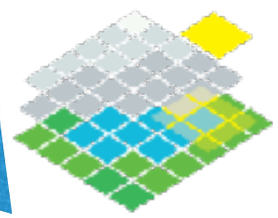


PPM_RC

Echtzeit-

Bildanalyse

Echtzeit-Darstellung



scenery

Simulation

OpenFPM

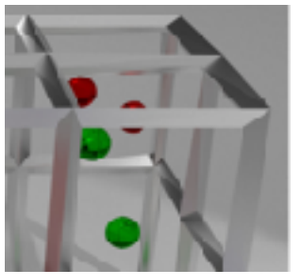
Photo: SZonline

Freie, offene Software!

Bildanalyse: MOSAICsuite for Fiji/ImageJ

- Segmentierung, Verbesserung, Tracking, Objektanalyse
- mosaic.mpi-cbg.de

Simulation: OpenFPM



- Skalierbare Hochleistungssimulationen
- openfpm.mpi-cbg.de

Programmierung: PPME

- Schnelle Softwareentwicklung
- bitbucket.org/ppme/

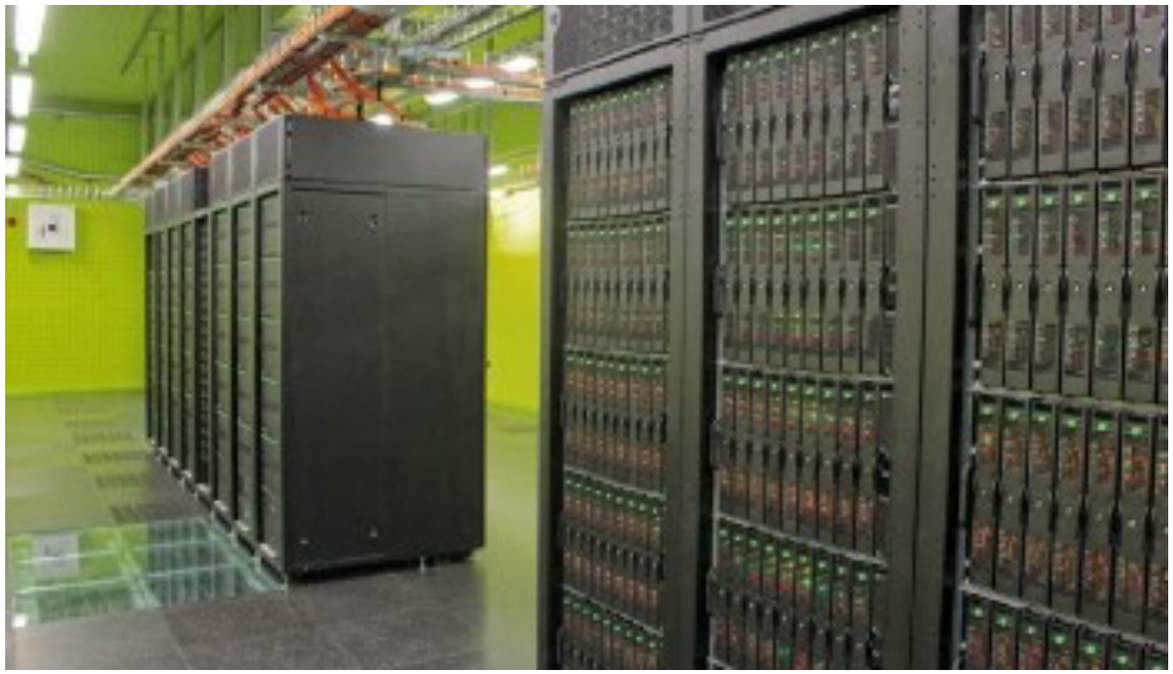
Echtzeitmikroskopie: ClearVolume

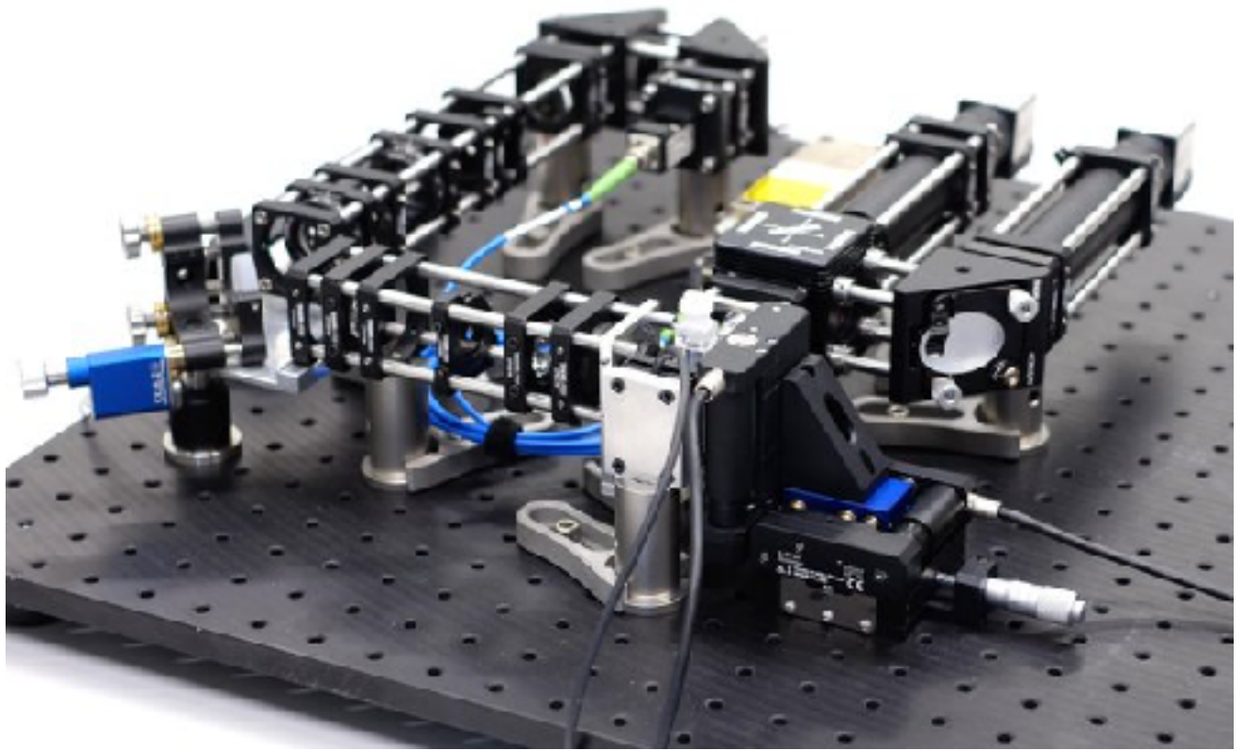
- Visualisierung während der Bildaufnahme
- https://github.com/clearvolume

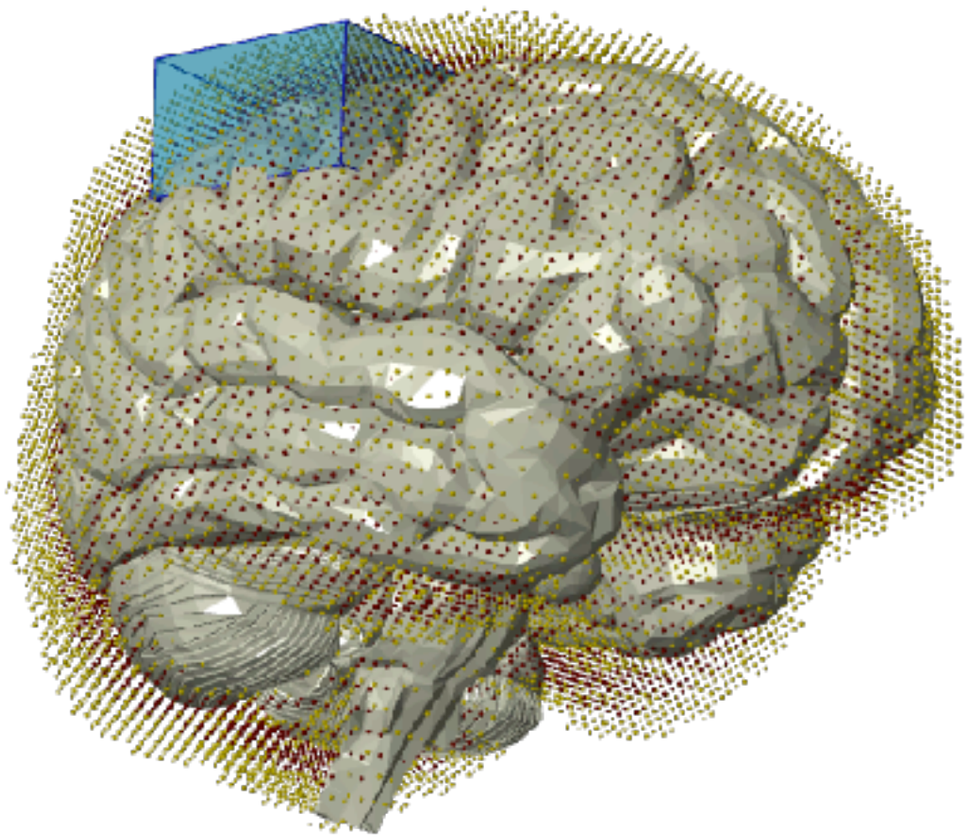
Virtuelle Realität: Scenery

- Immersive Mikroskopie mit Nutzerinteraktion
- https://github.com/clearvolume/scenery

Cool Stuff!







Research Topics @MOSAICgroup1

- Scientific Computing: multiscale numerical simulation algorithms that can handle biological complexity
  —> **Particle Methods**
- Computer Graphics (with Prof. Gumhold and Prof. Dachselt): real-time VR of distributed 3D microscopy and simulation data
- Computer Vision: bio-image segmentation using particles, motion tracking
- High-Performance Computing: PPM Library, OpenFPM
- Programming languages and Compilers (with Prof. Castrillon): PPML
- Theory (with Prof. Baader): Automata and information theory of cellular and biochemical information processing

center for
systems biology
dresden

mosaic.mpi-cbg.de