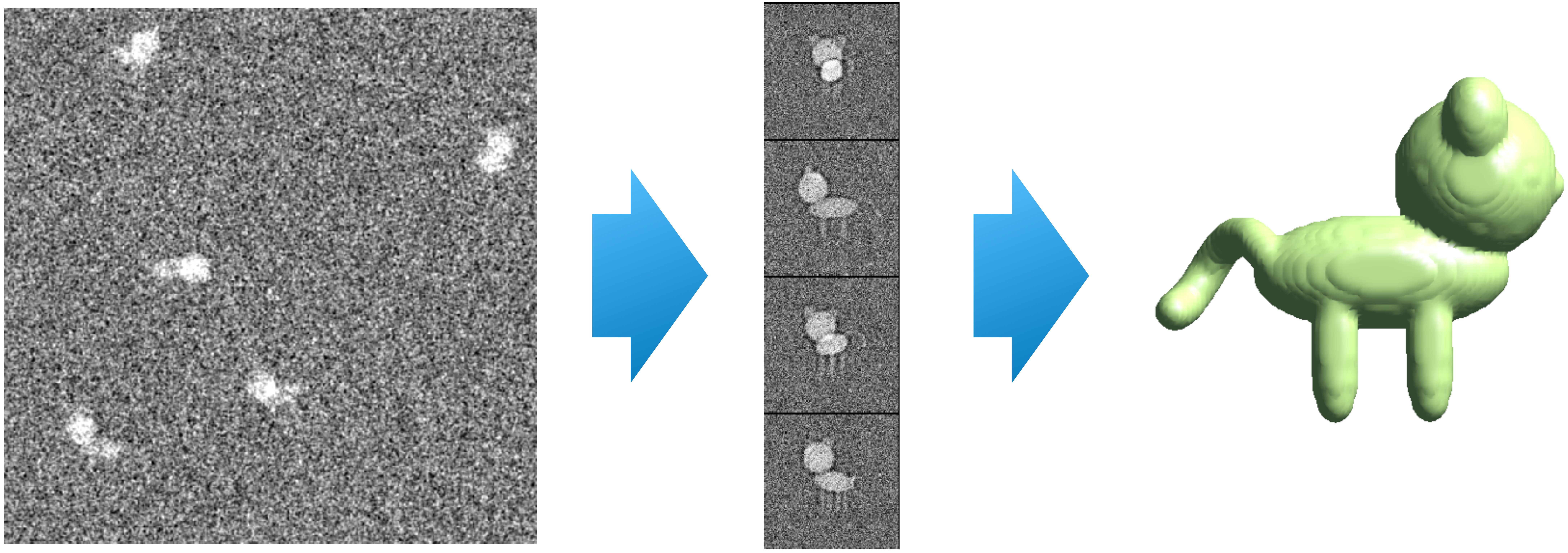
A machine learning method for resolving heterogeneity in CryoEM single particle reconstruction

Muyuan Chen

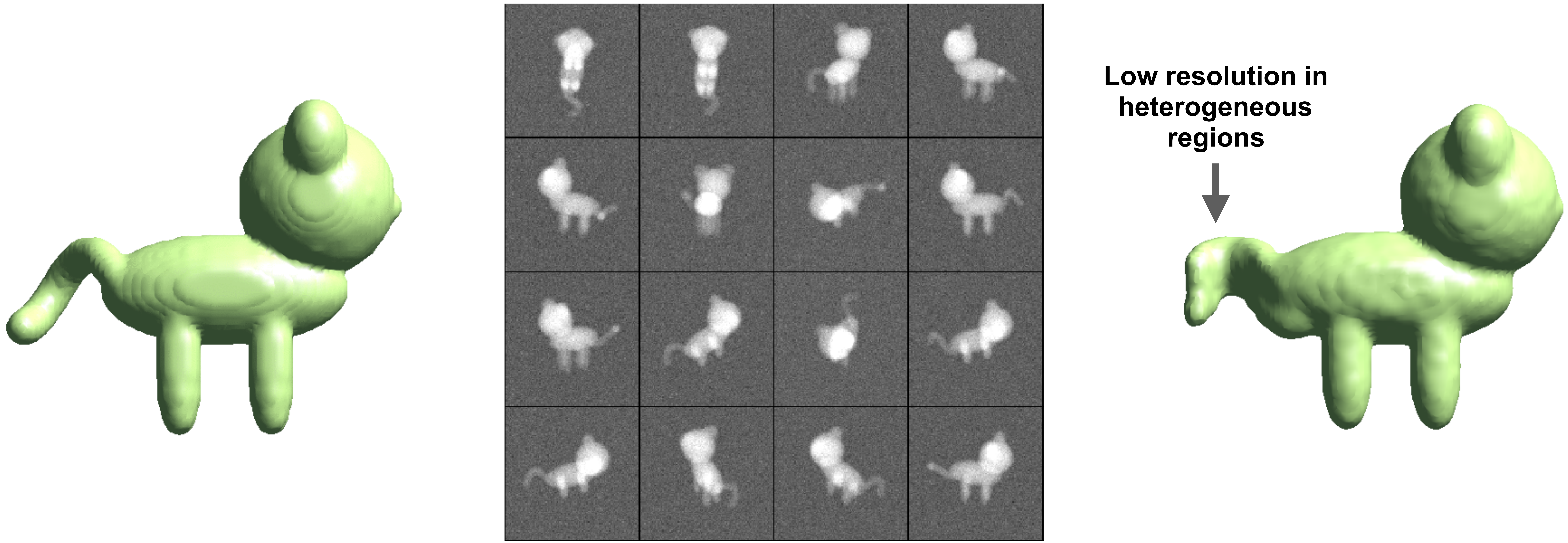
2018-03

Baylor College of Medicine

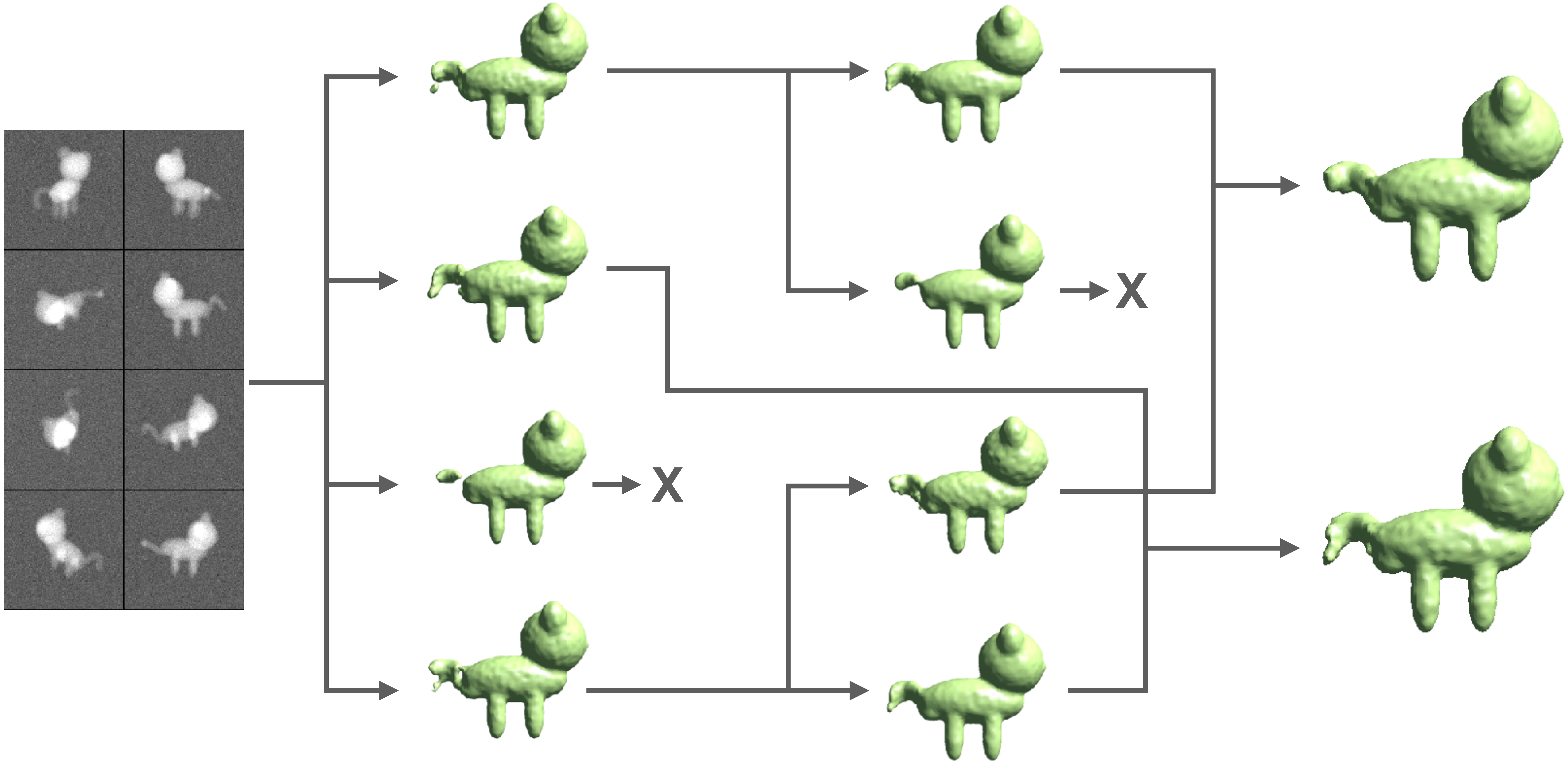
Single particle reconstruction



Heterogeneity in Single particle reconstruction



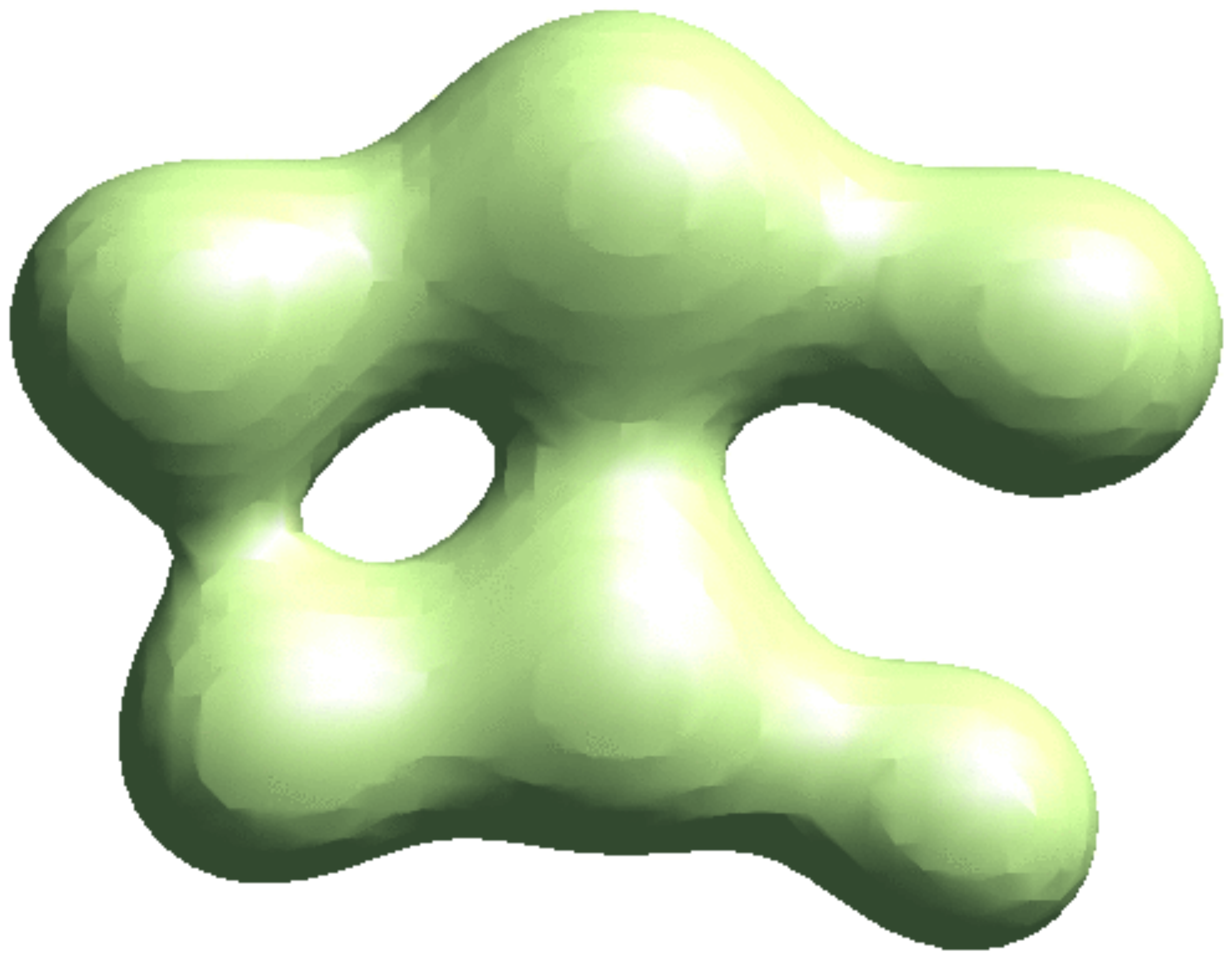
Classical strategy: multi-model refinement



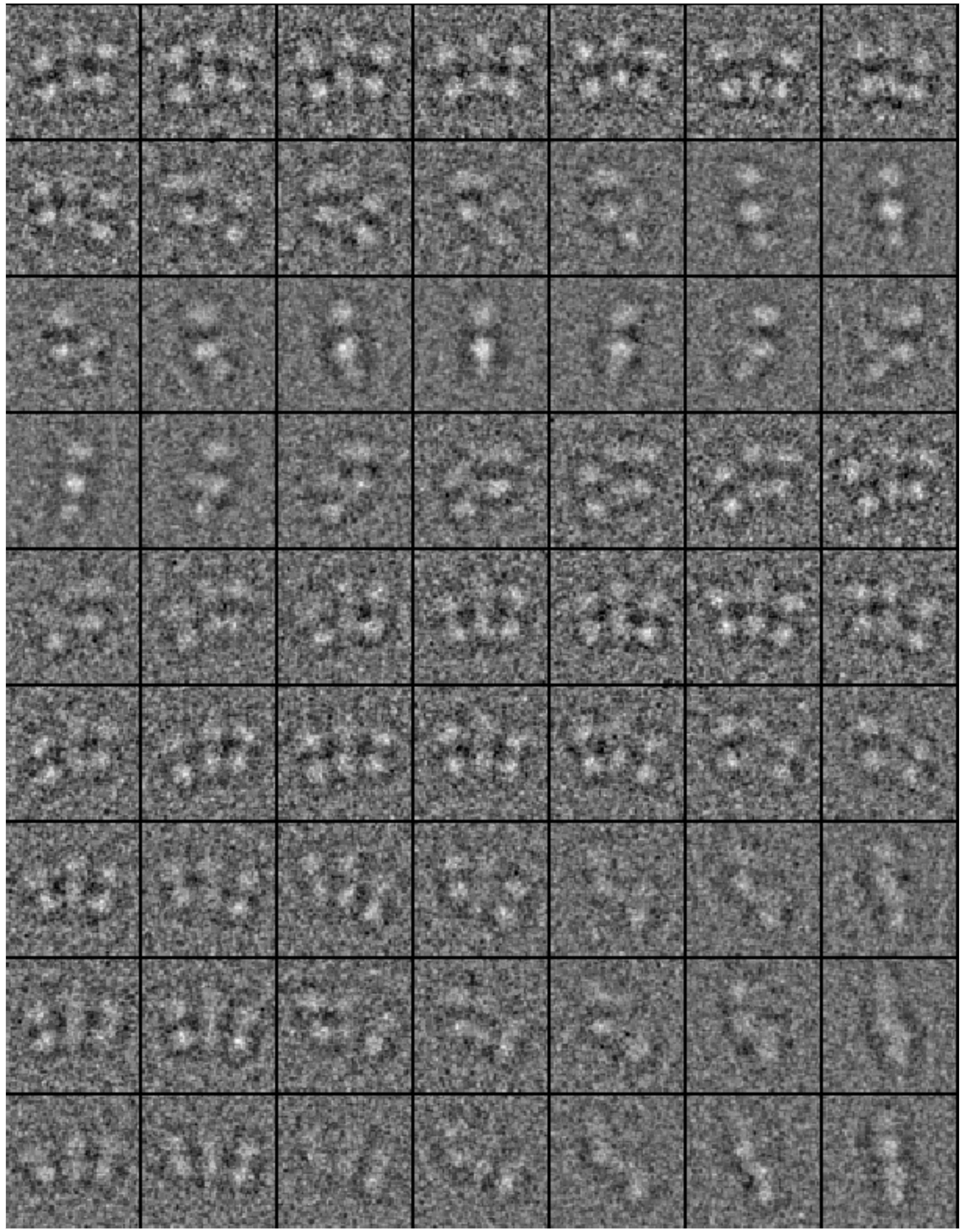
Works, but is suboptimal…

Determination of conformational landscape

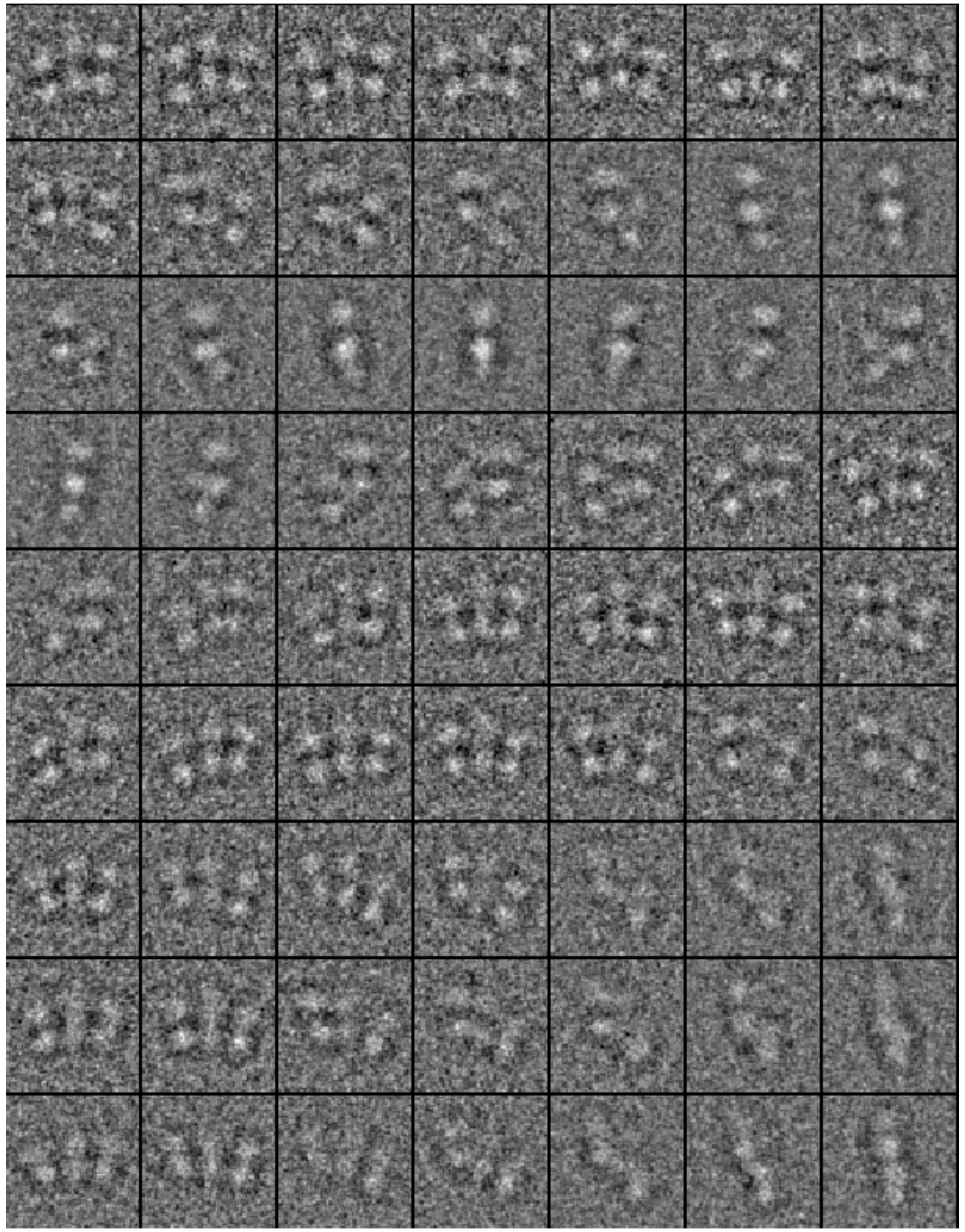
A even simpler simulated dataset

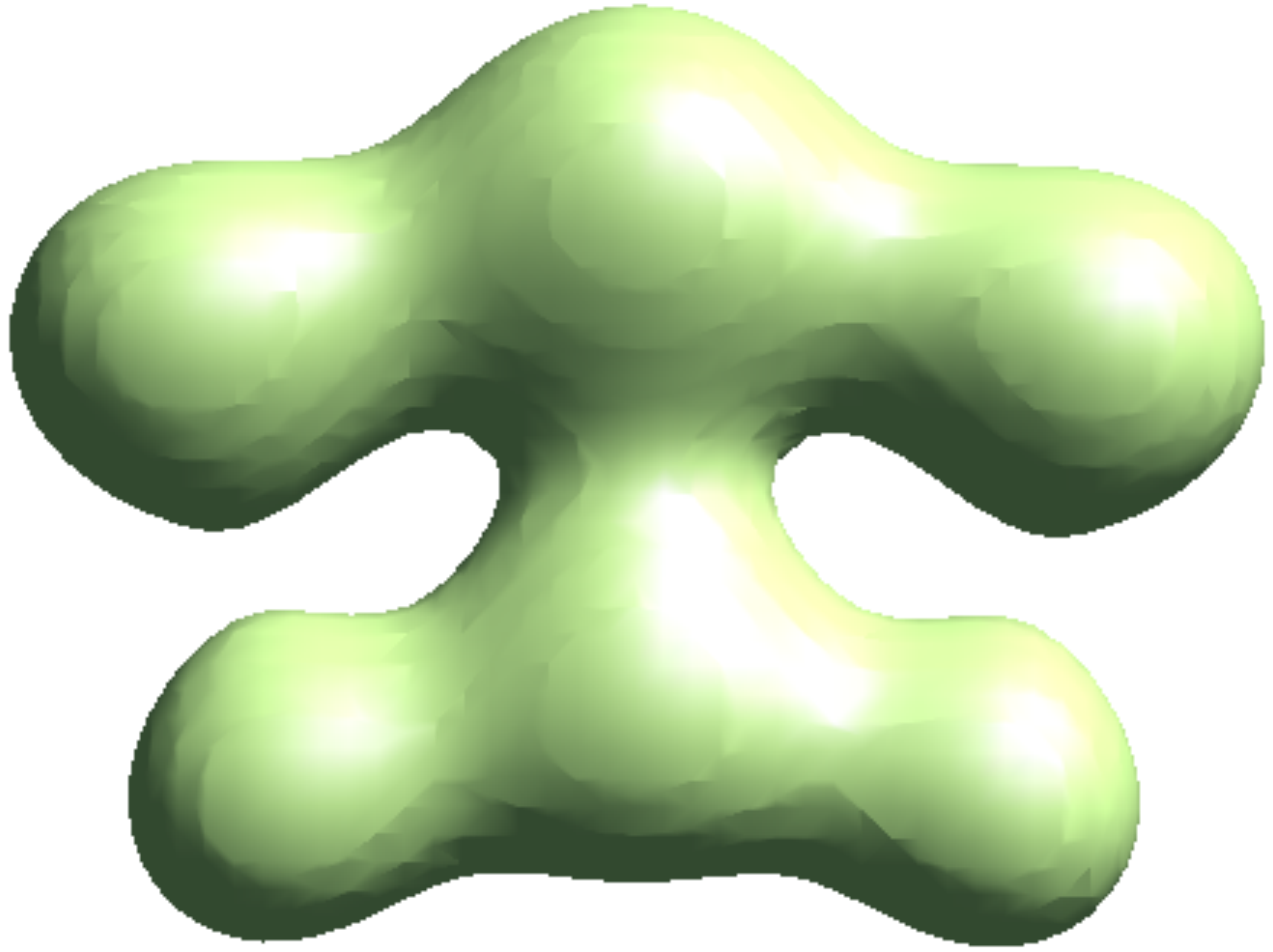


- 3000 particles from 11 classes
- Low resolution SNR: ~1
- 2 - 3.5um defocus

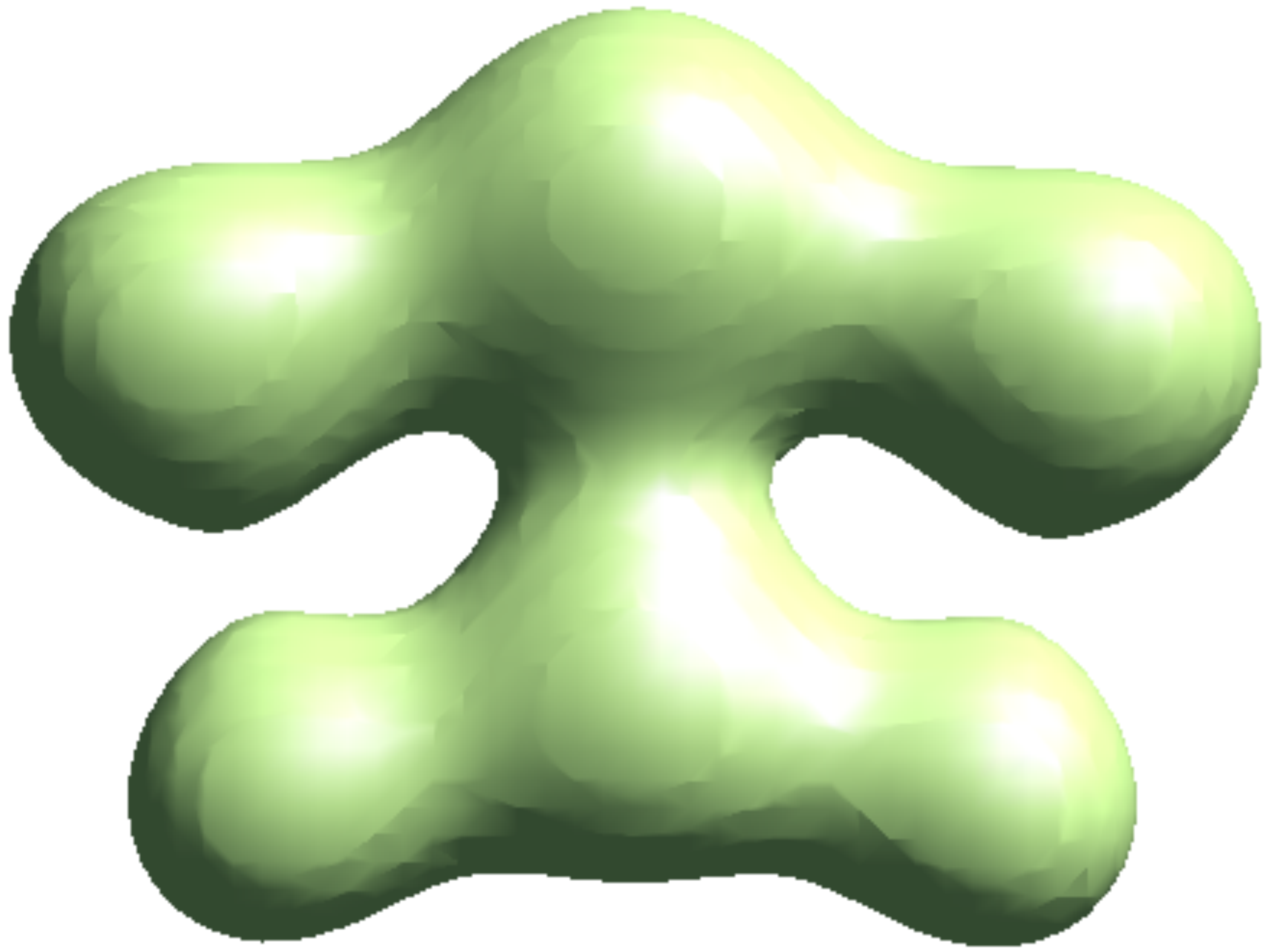


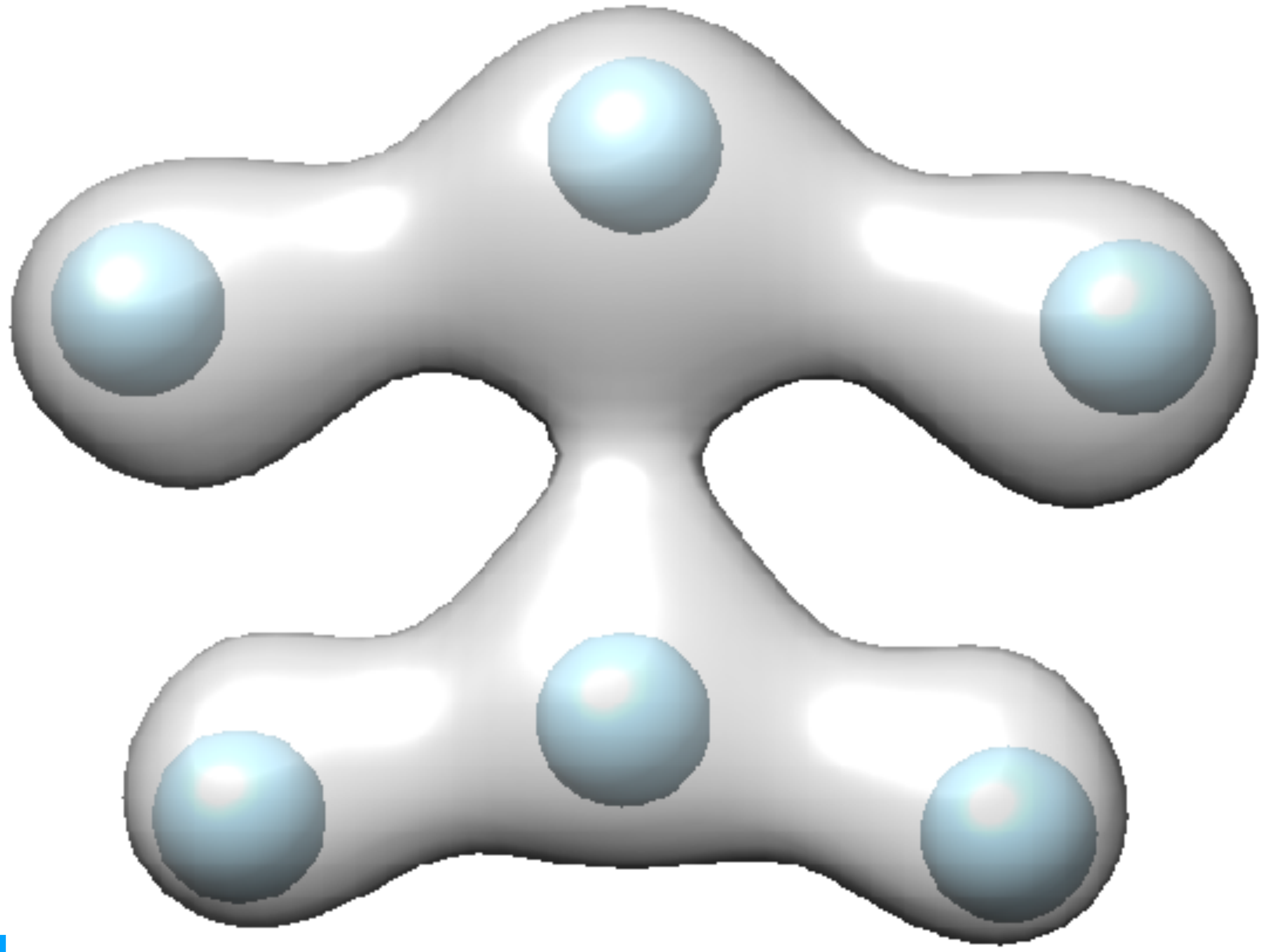
Start form a single model refinement





Gaussian representation of density map





$$map(\mathbf{x}) = \sum_{j=0}^{n} A_j e^{-\frac{\mathbf{x}-\mathbf{p}_j}{\sigma_j^2}}$$

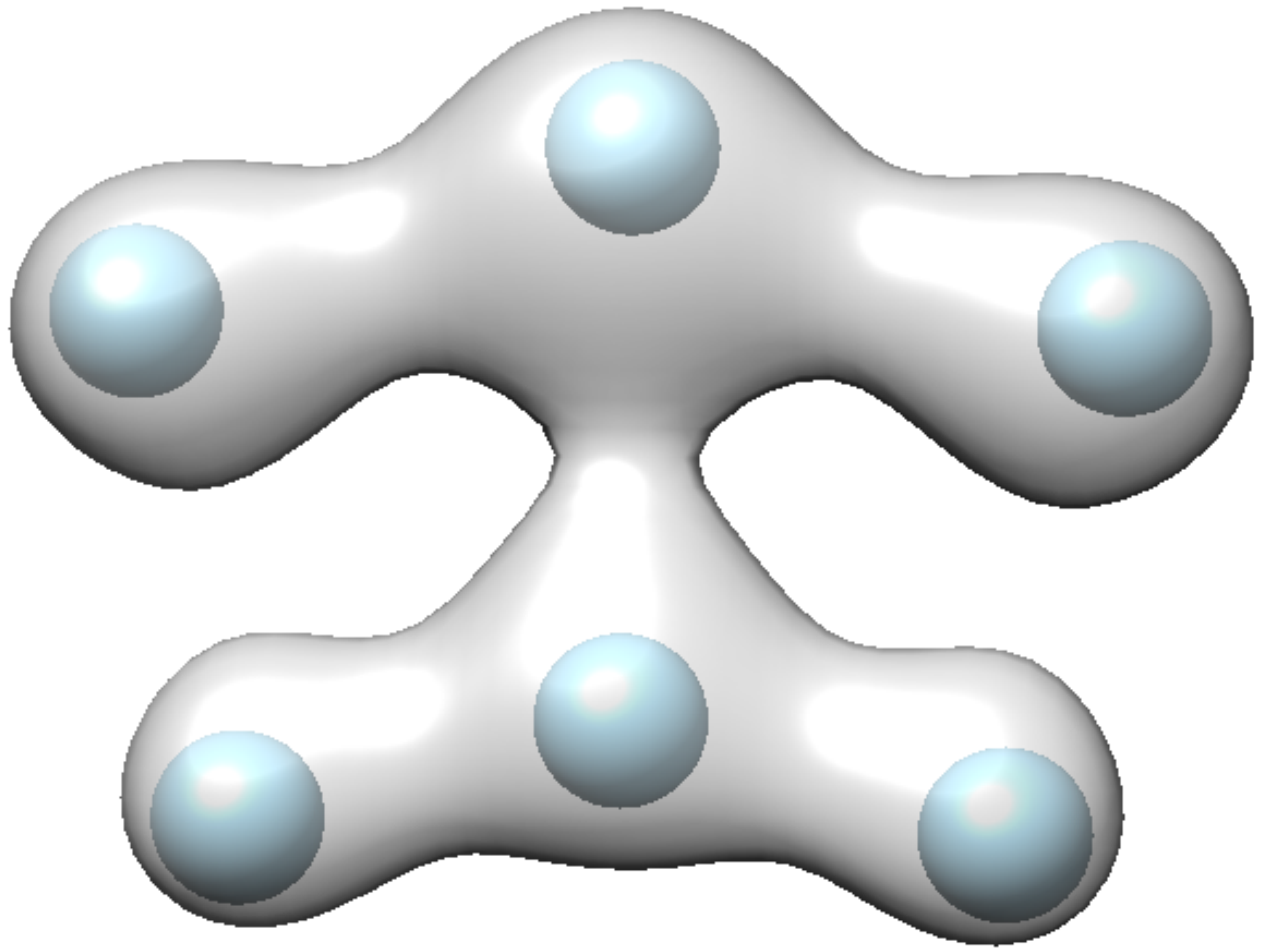
3D Position vector

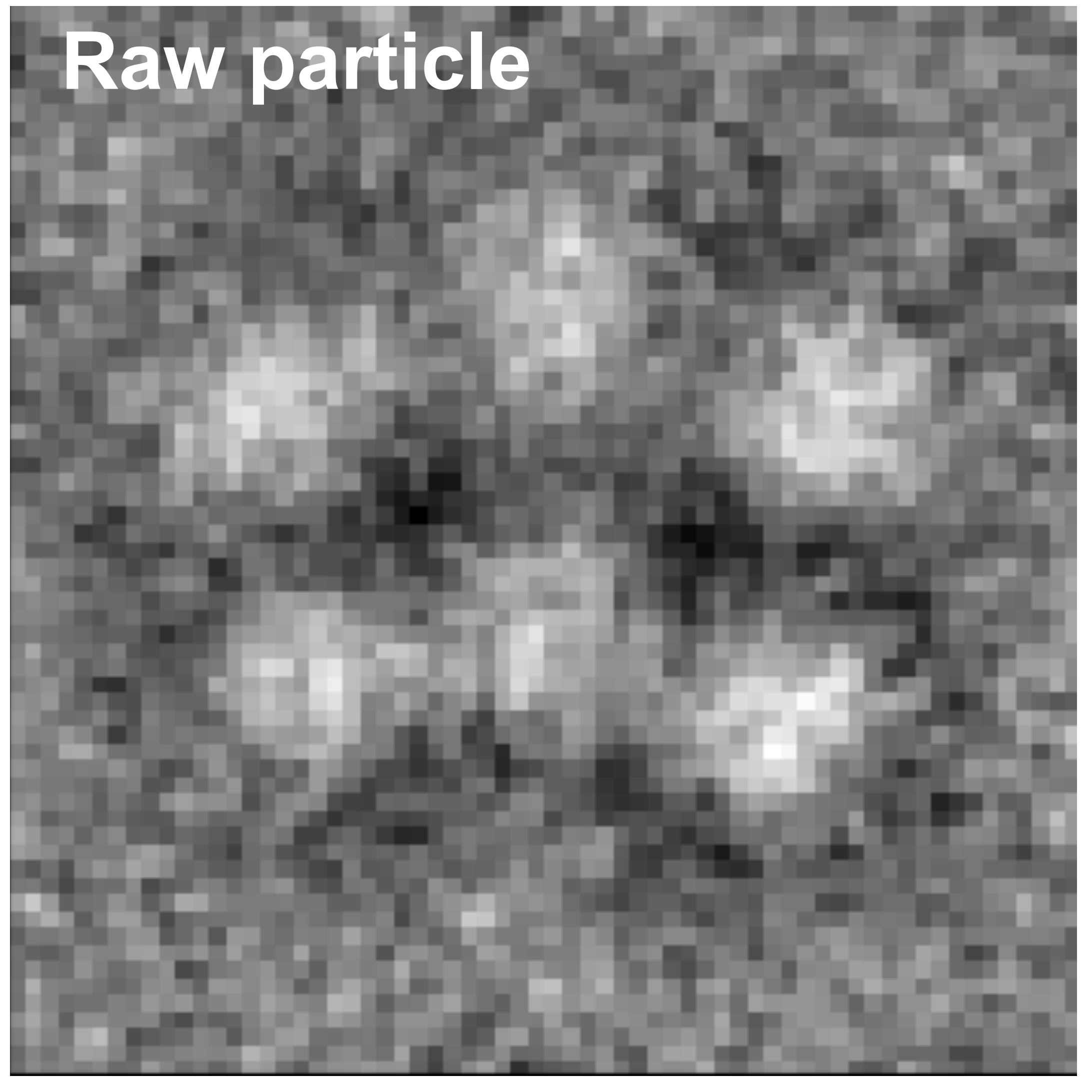
Amplitude

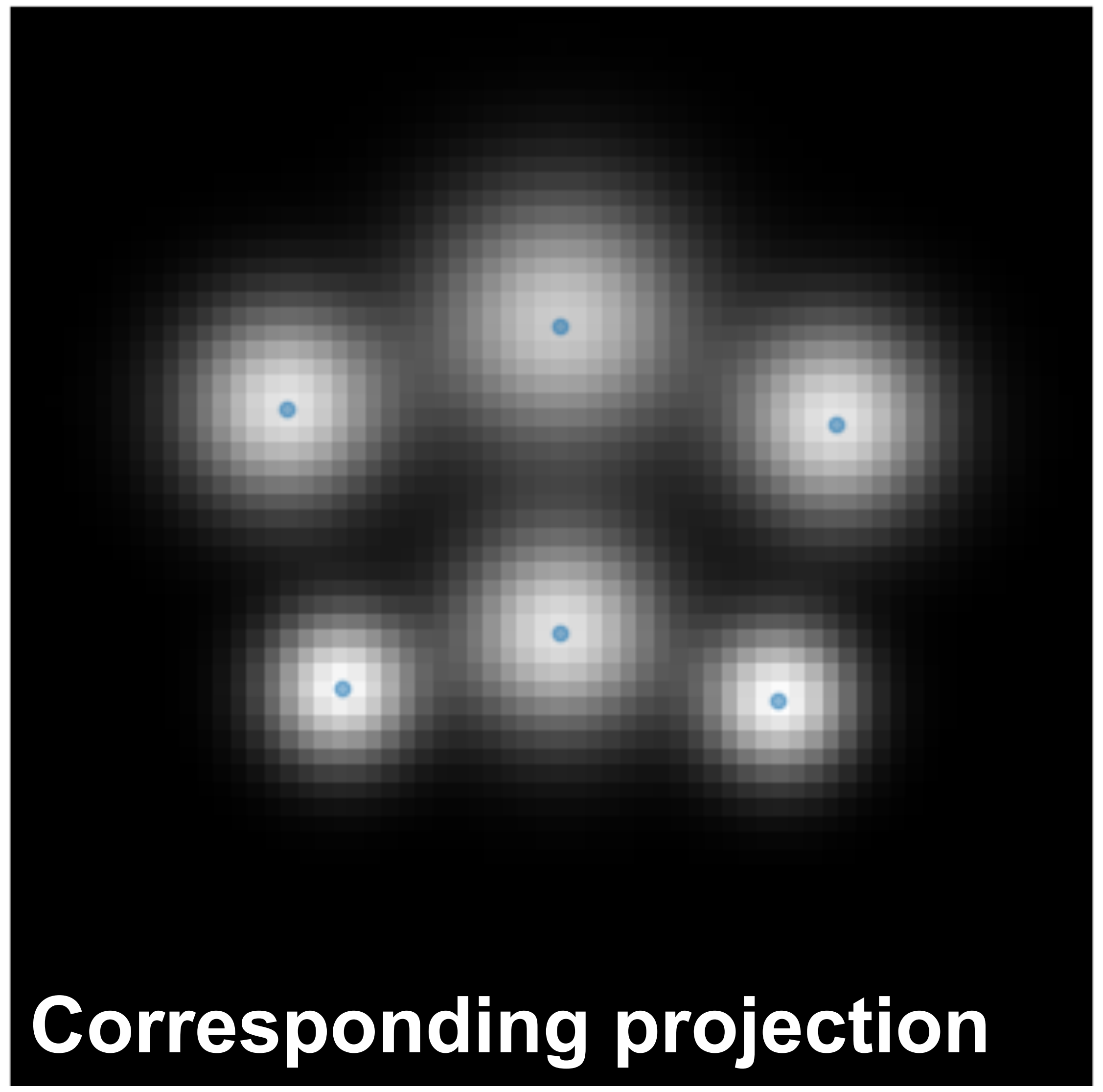
Sigma

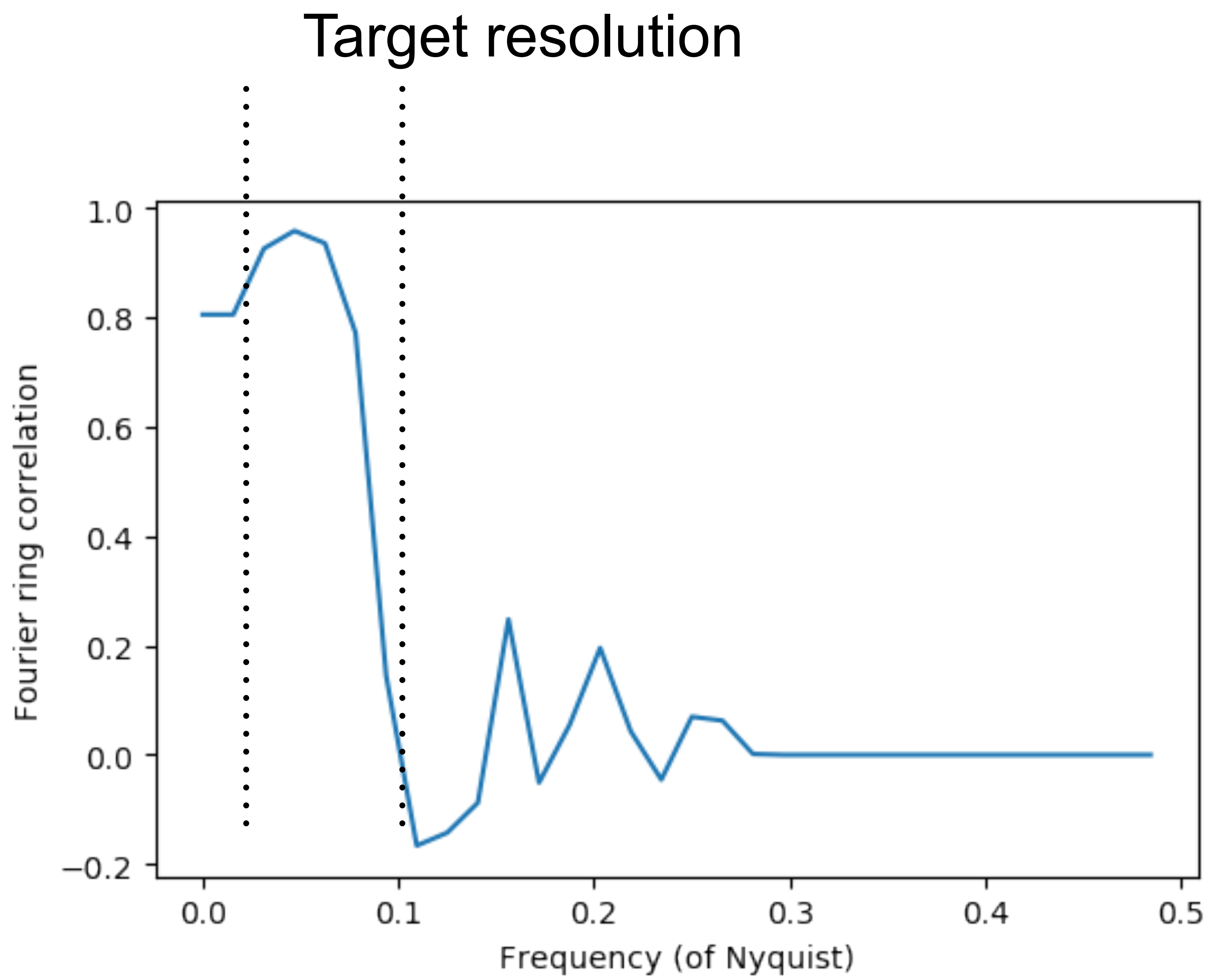
- Reduce complexity
- Easier to model continuous motion

Particle-projection comparison



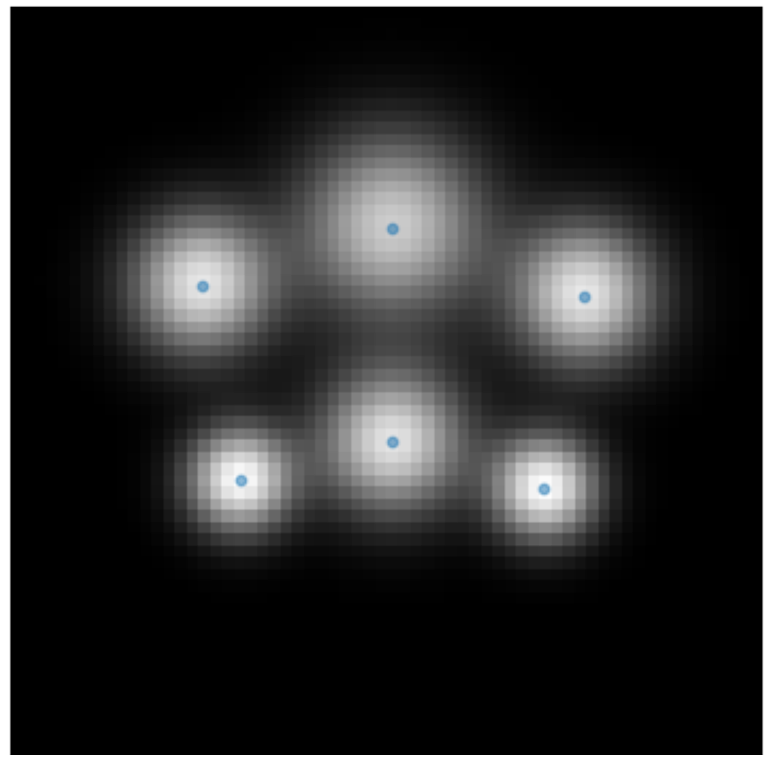


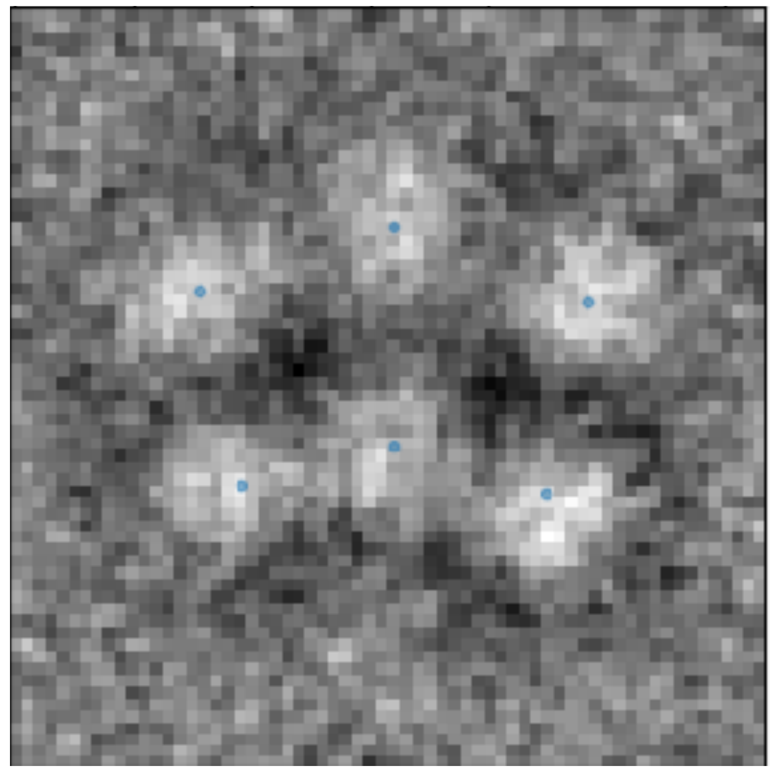


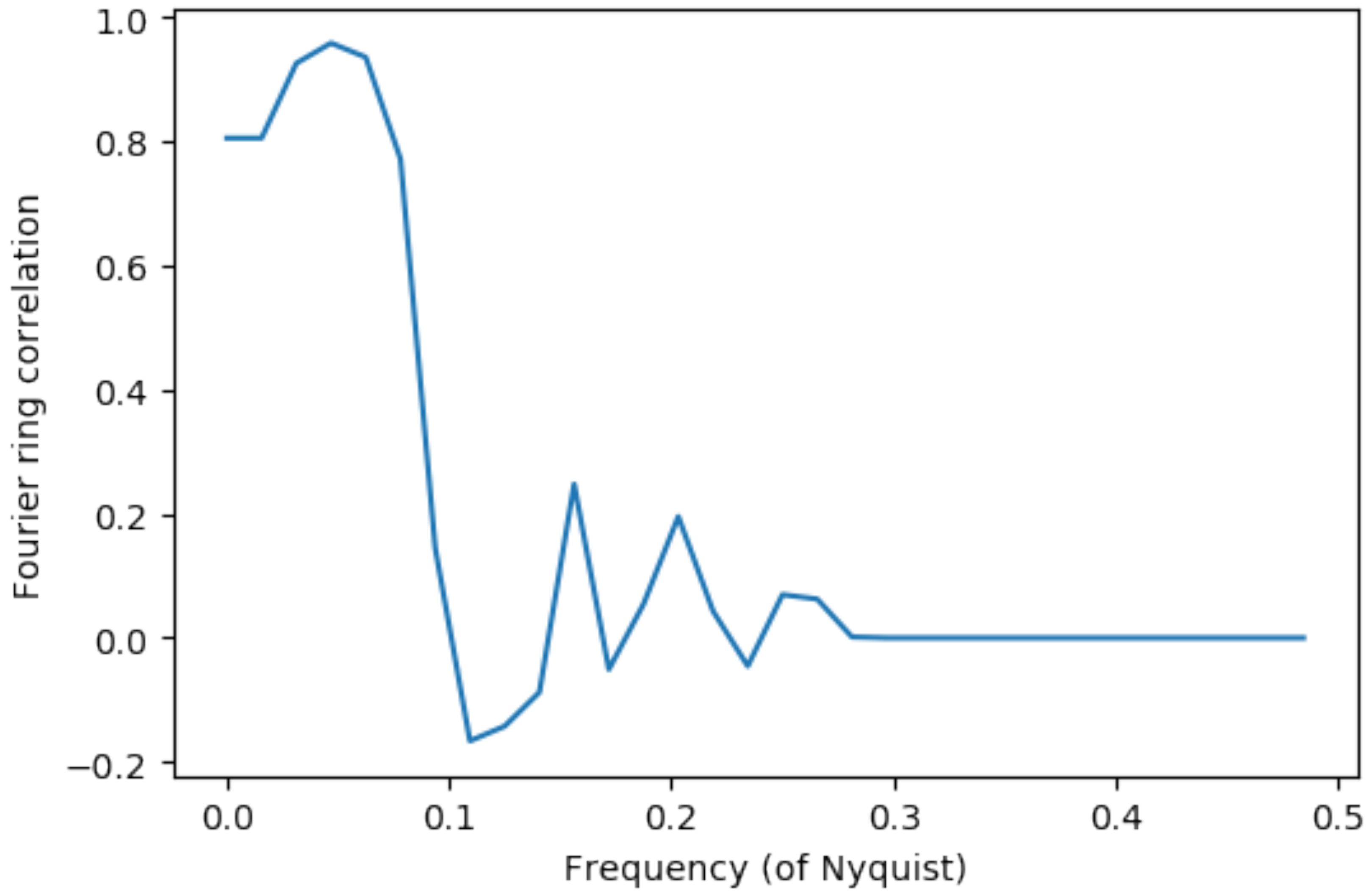


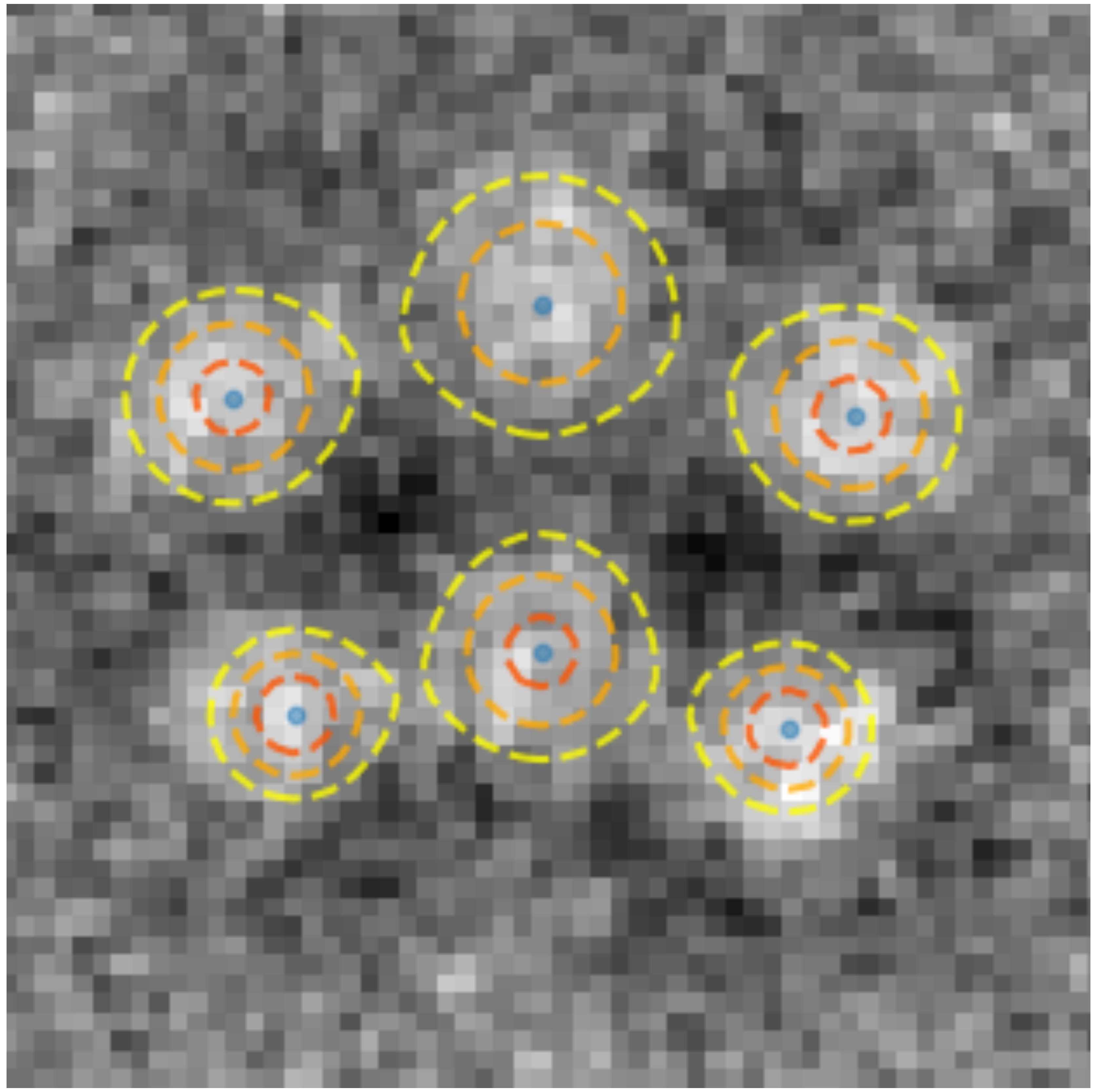
Particle-projection FRC

Gradient calculation





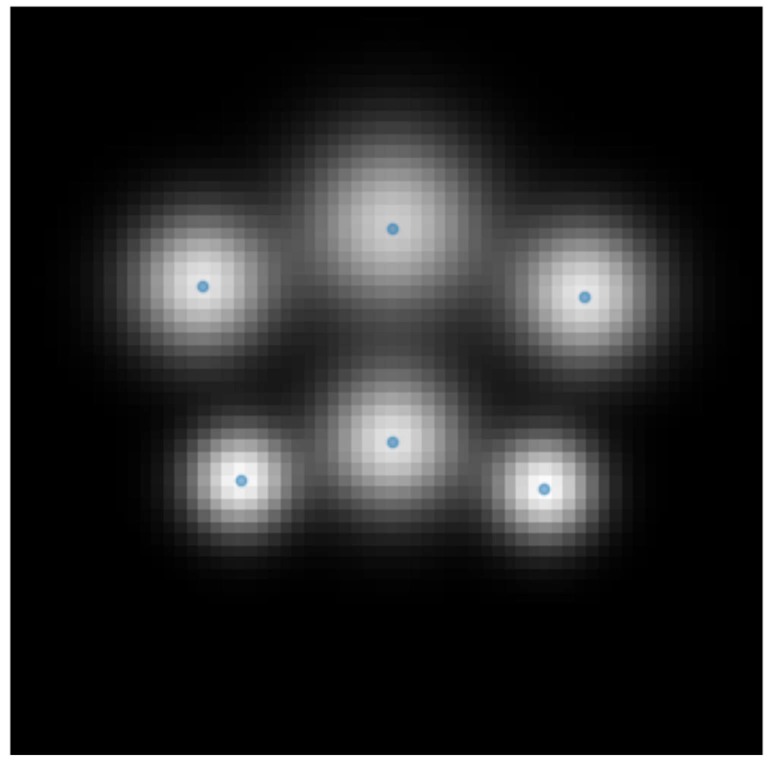


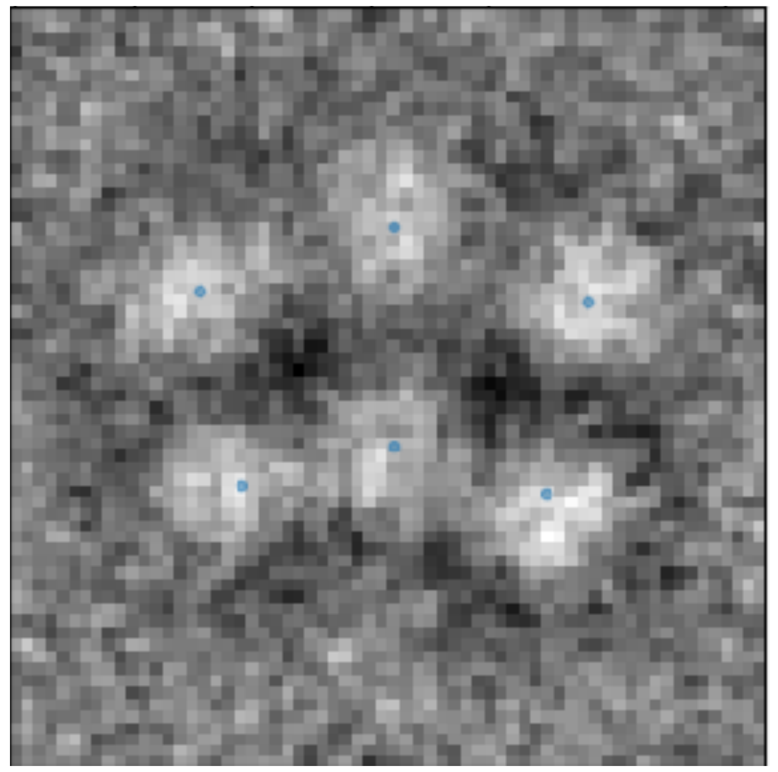


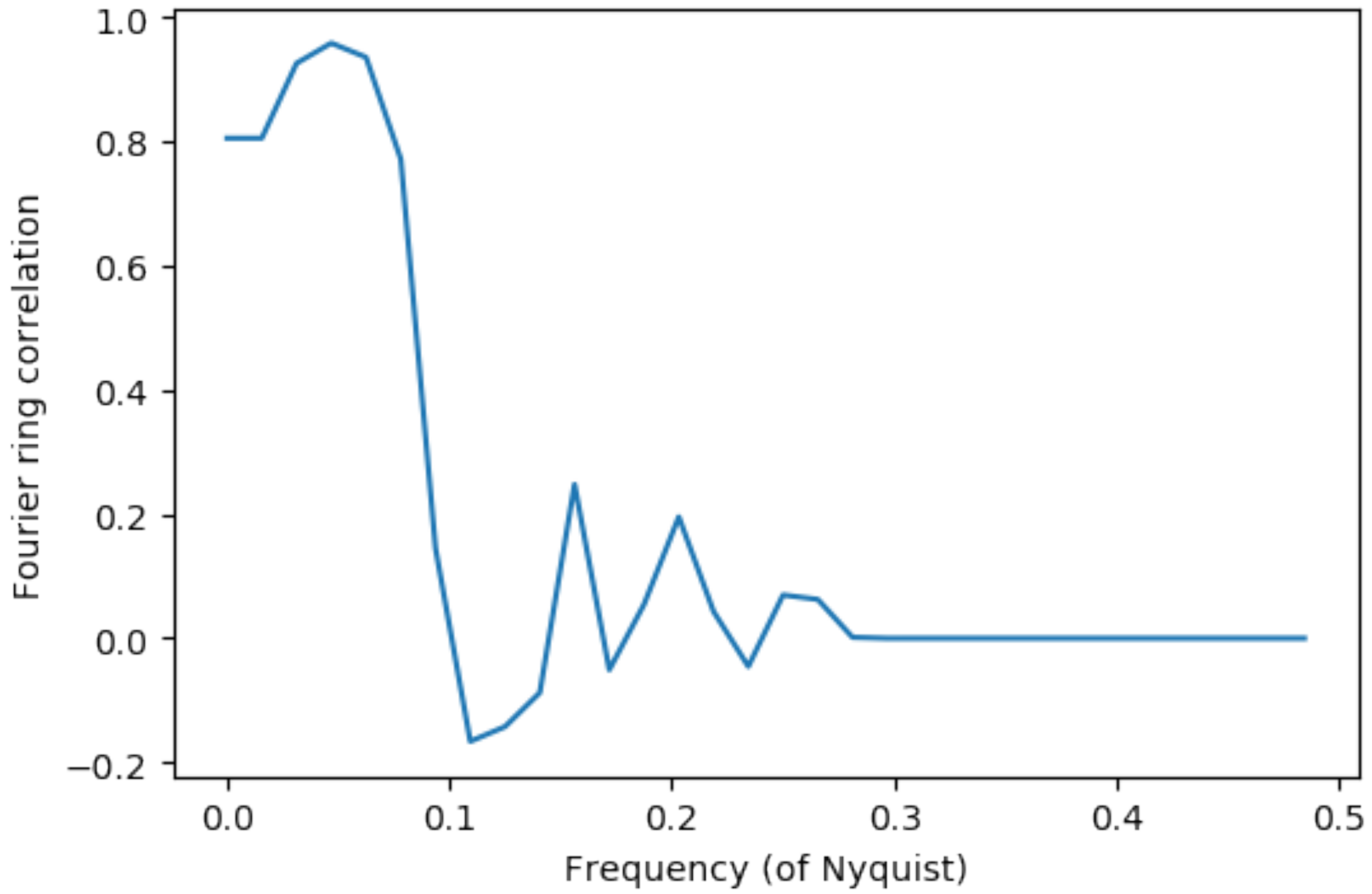
Goal:

To improve particle-projection matching (FRC)

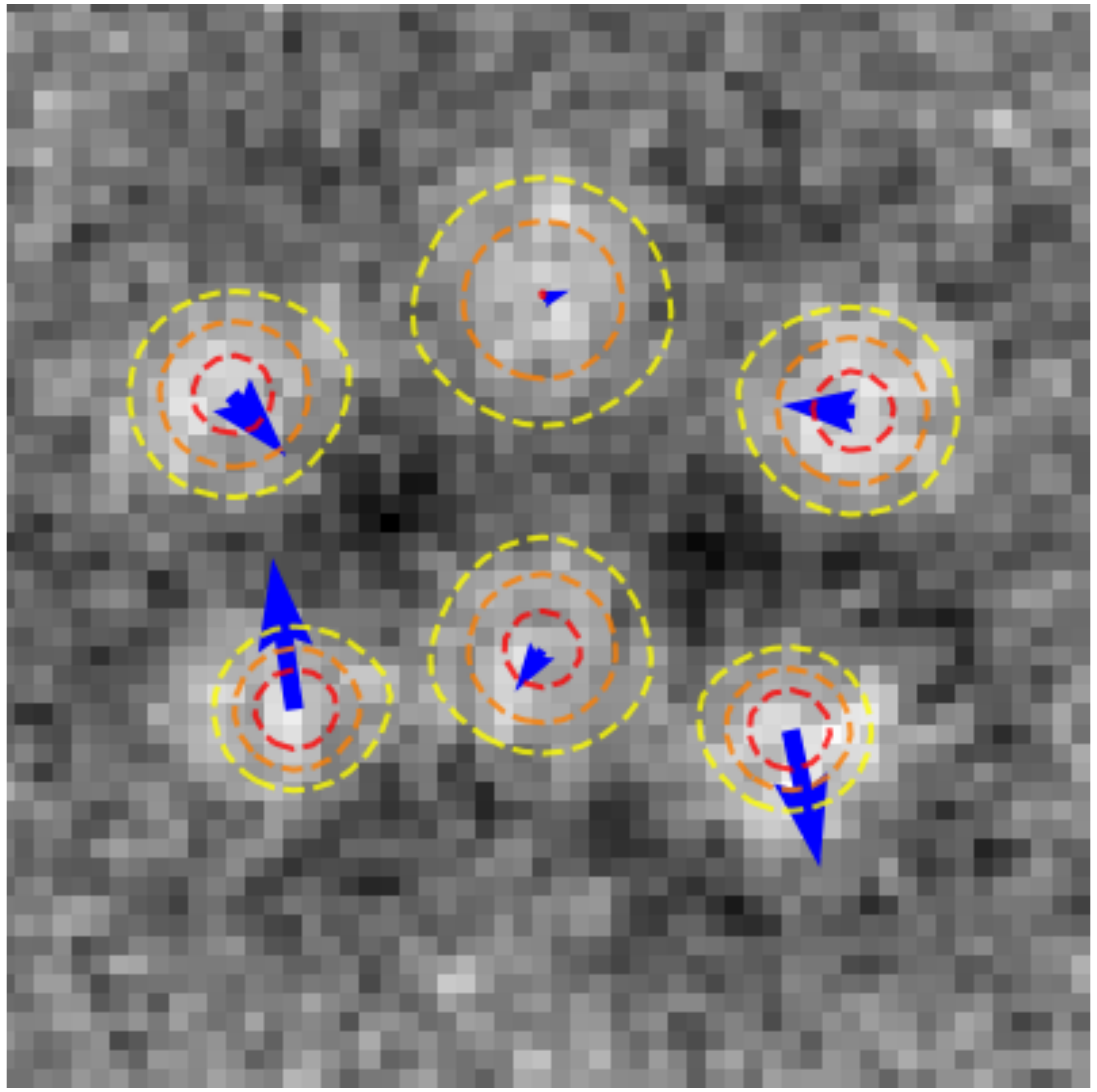
Gradient calculation







$$\mathbf{Grad}_j = \frac{\partial FRC}{\partial \mathbf{p}_j}$$

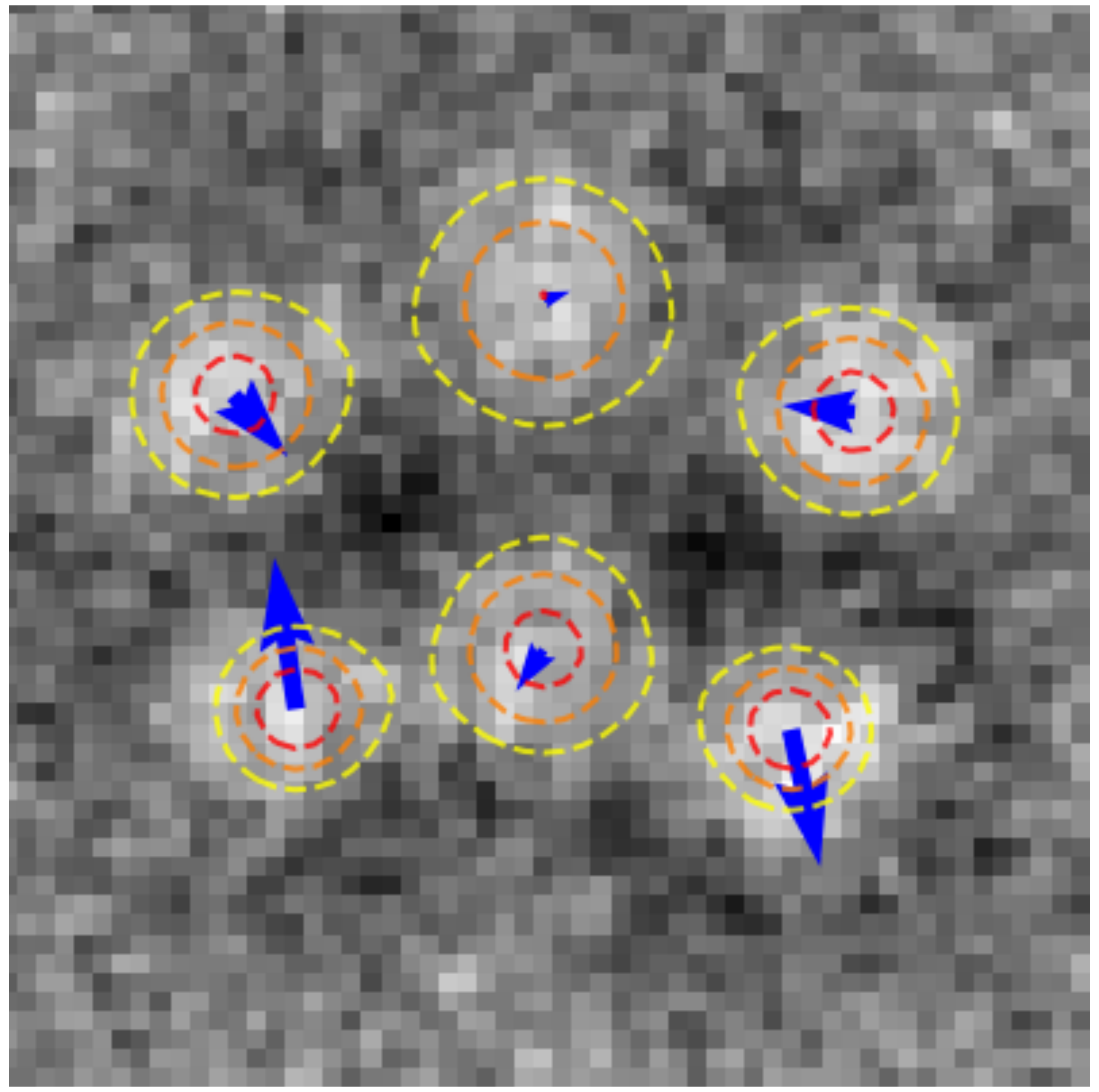


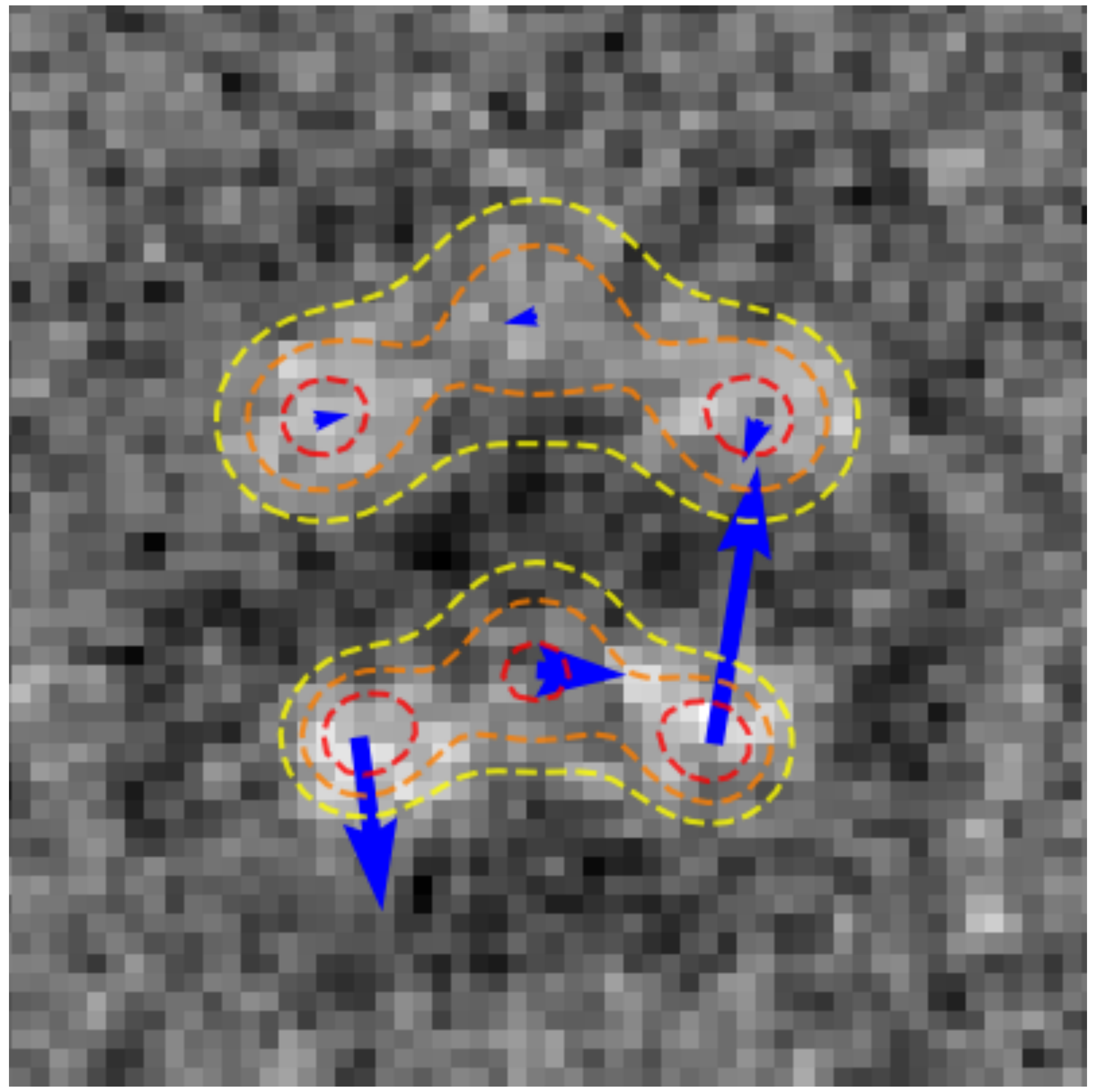
Goal:

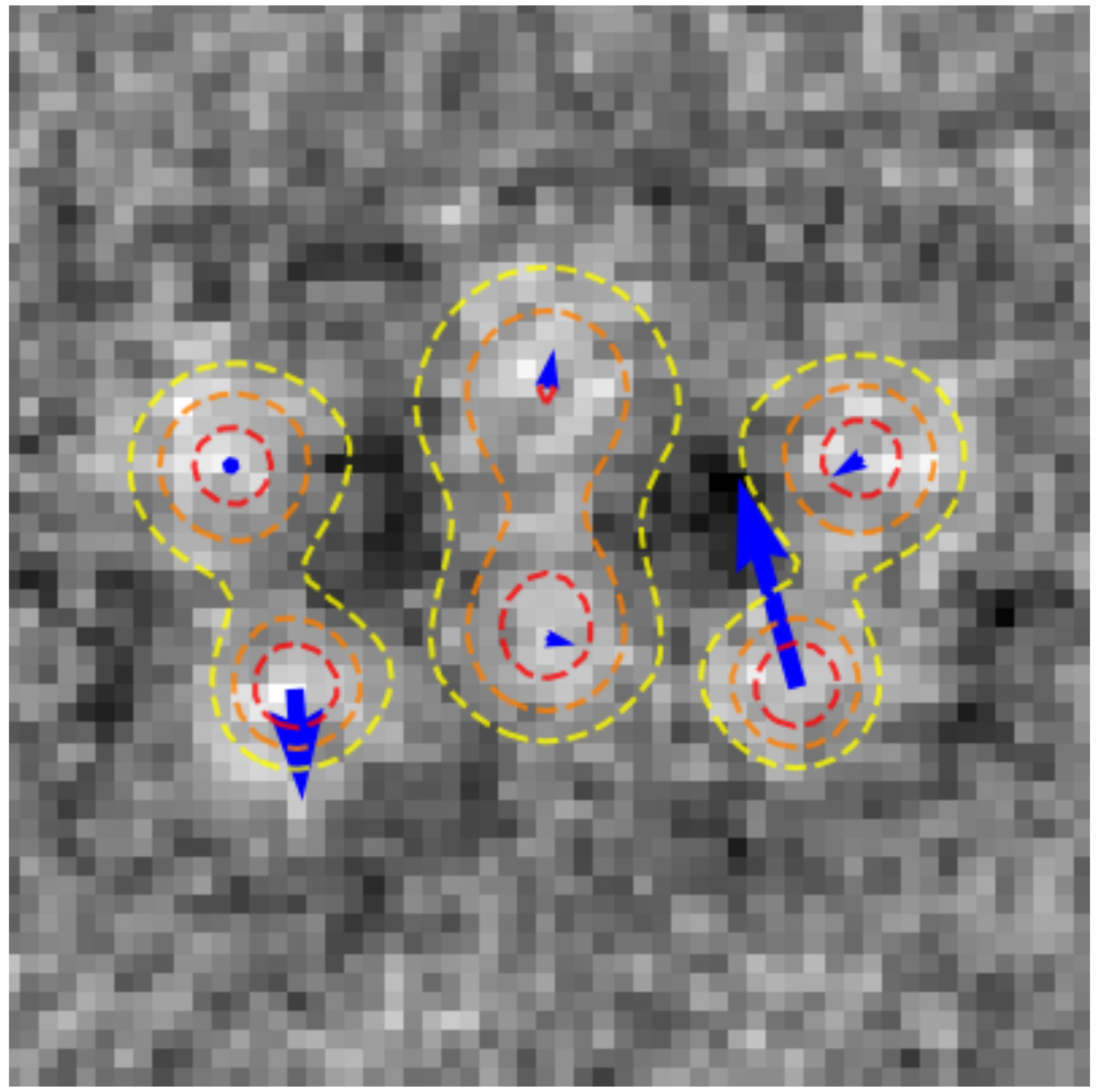
To improve particle-projection matching (FRC)

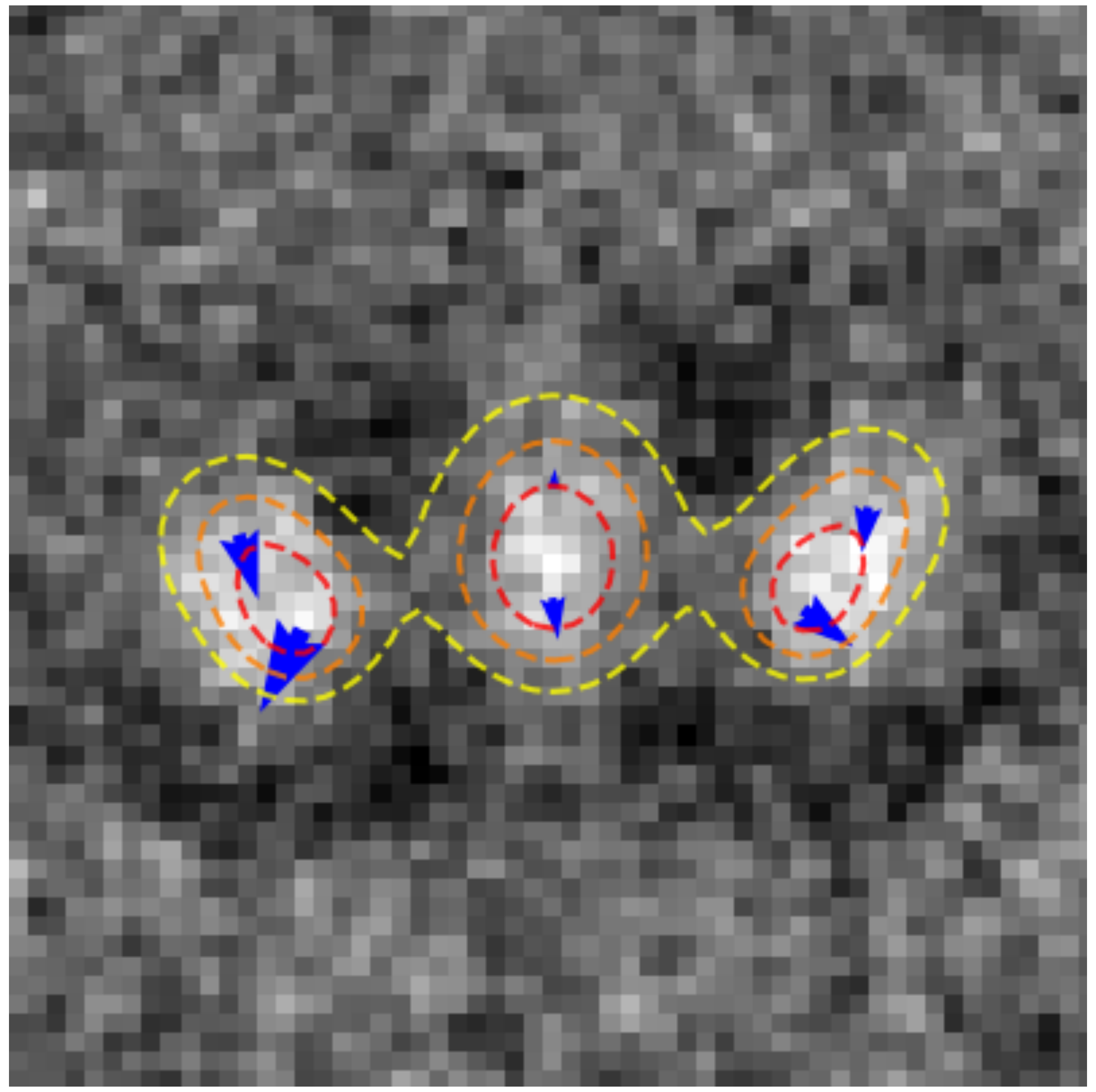
For each Gaussian, which direction should it move to improve the particle-projection FRC on each particle?

Gradient per particle per Gaussian



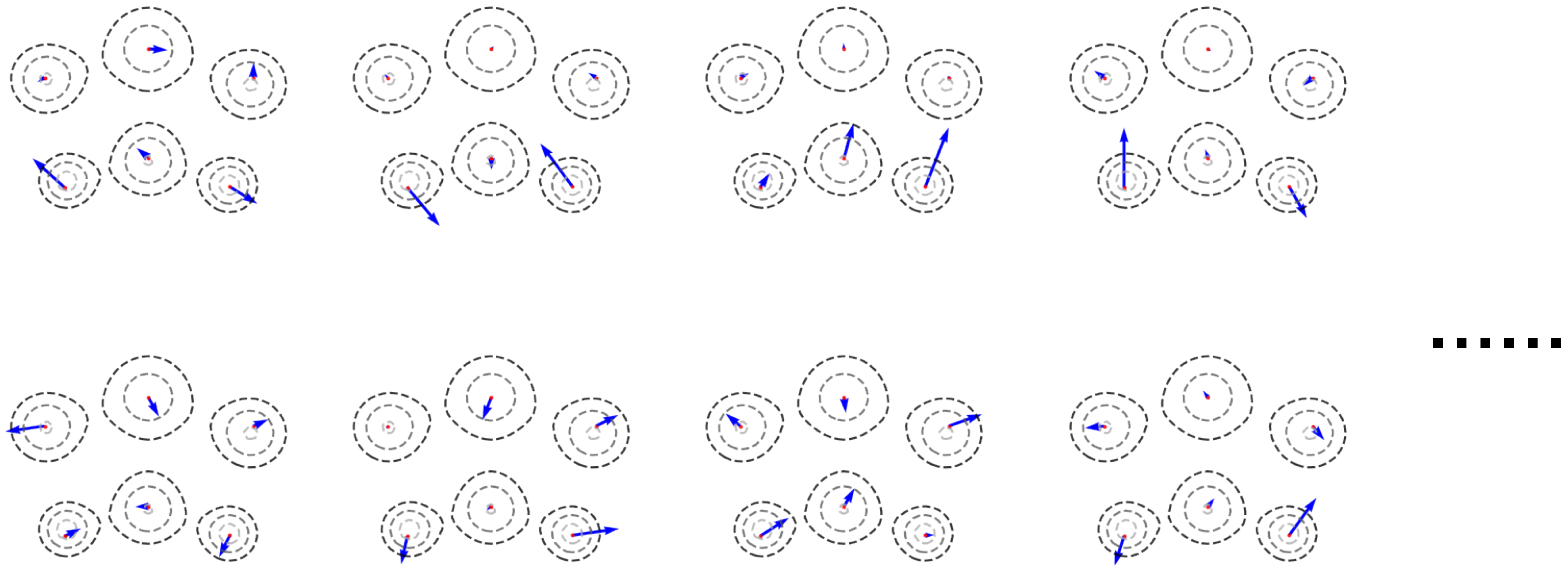






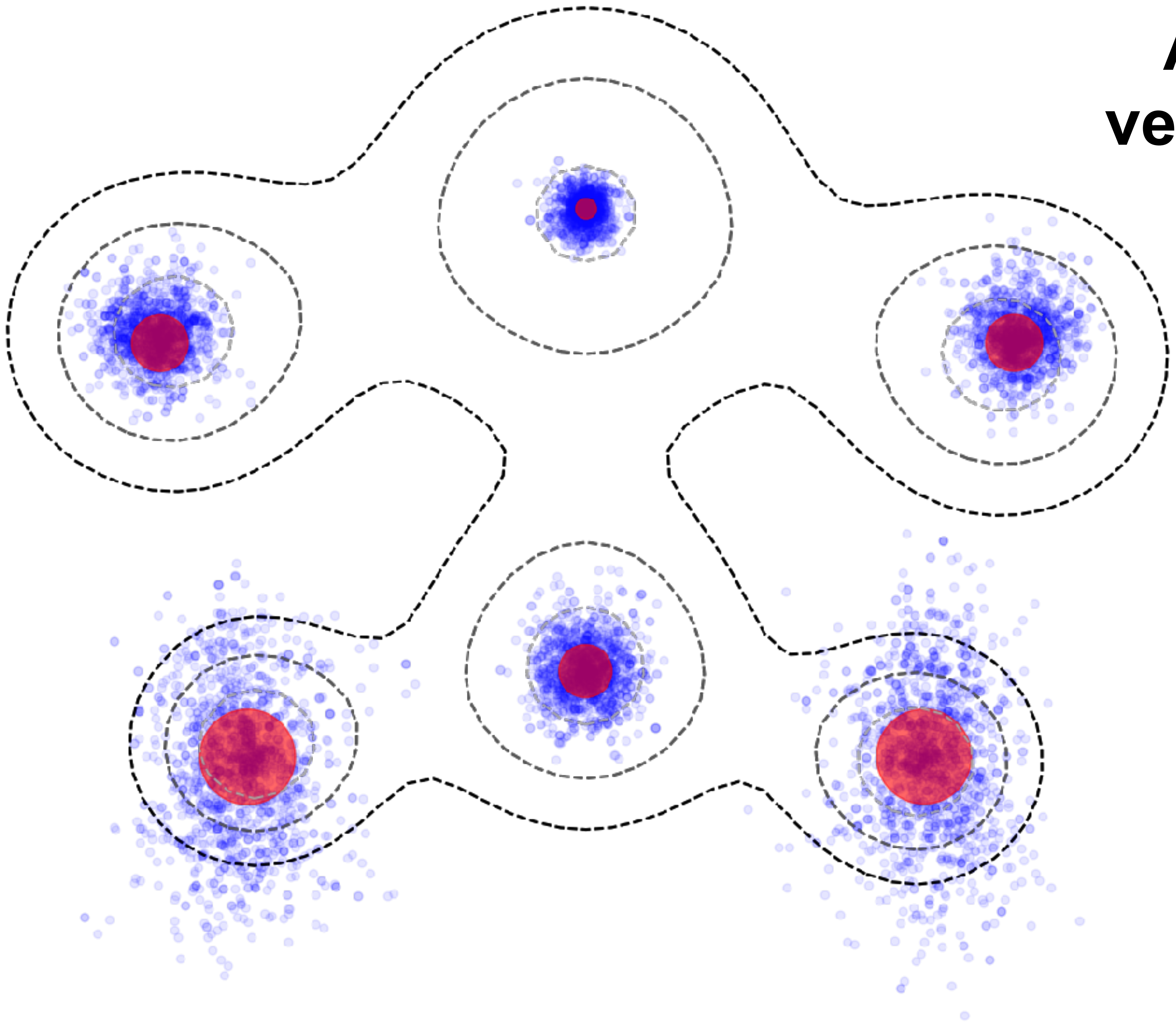
2D gradient vectors on projection plane

Gather gradient vectors of all particles in 3D



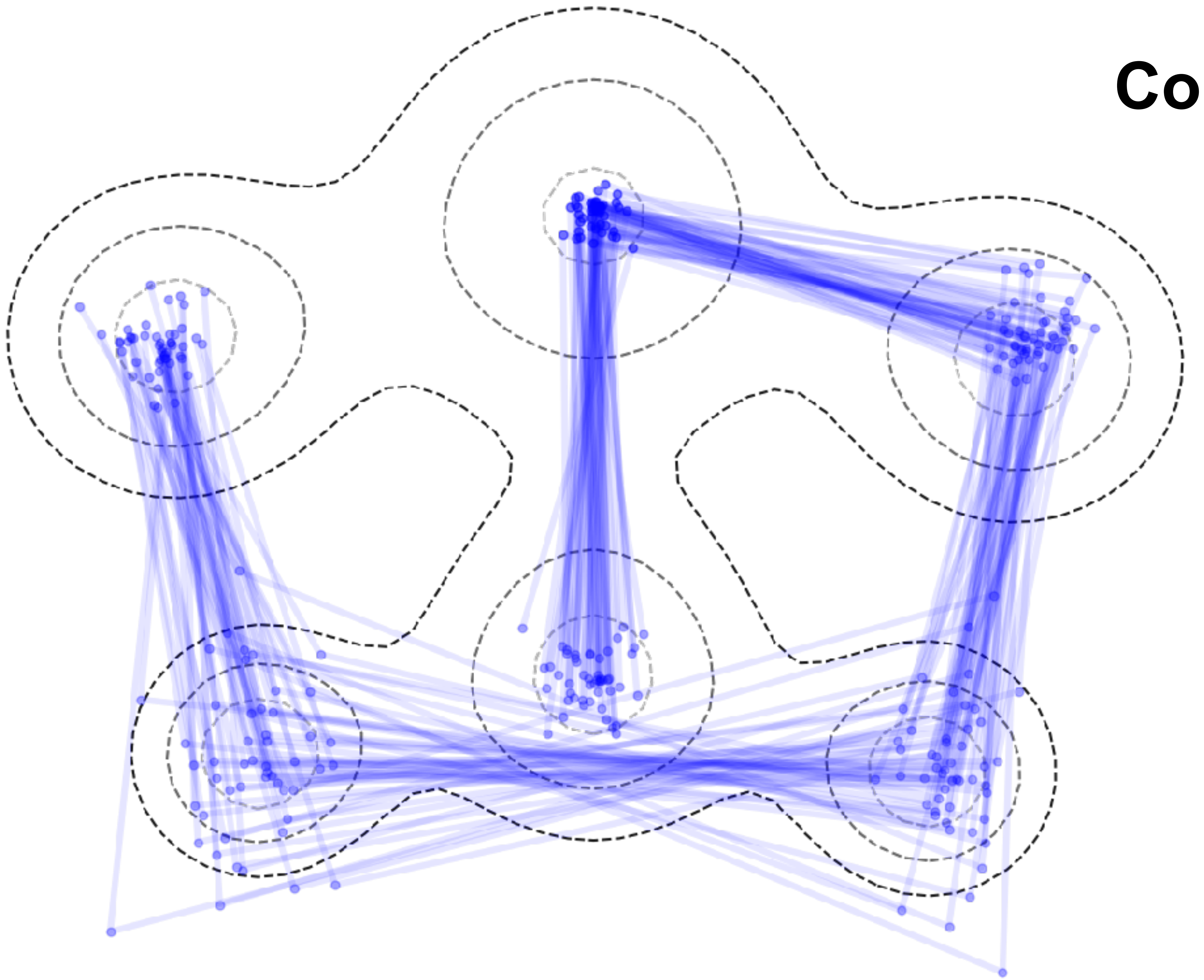
3D gradient vectors projected on x-y plane

Statistics on gradient vectors



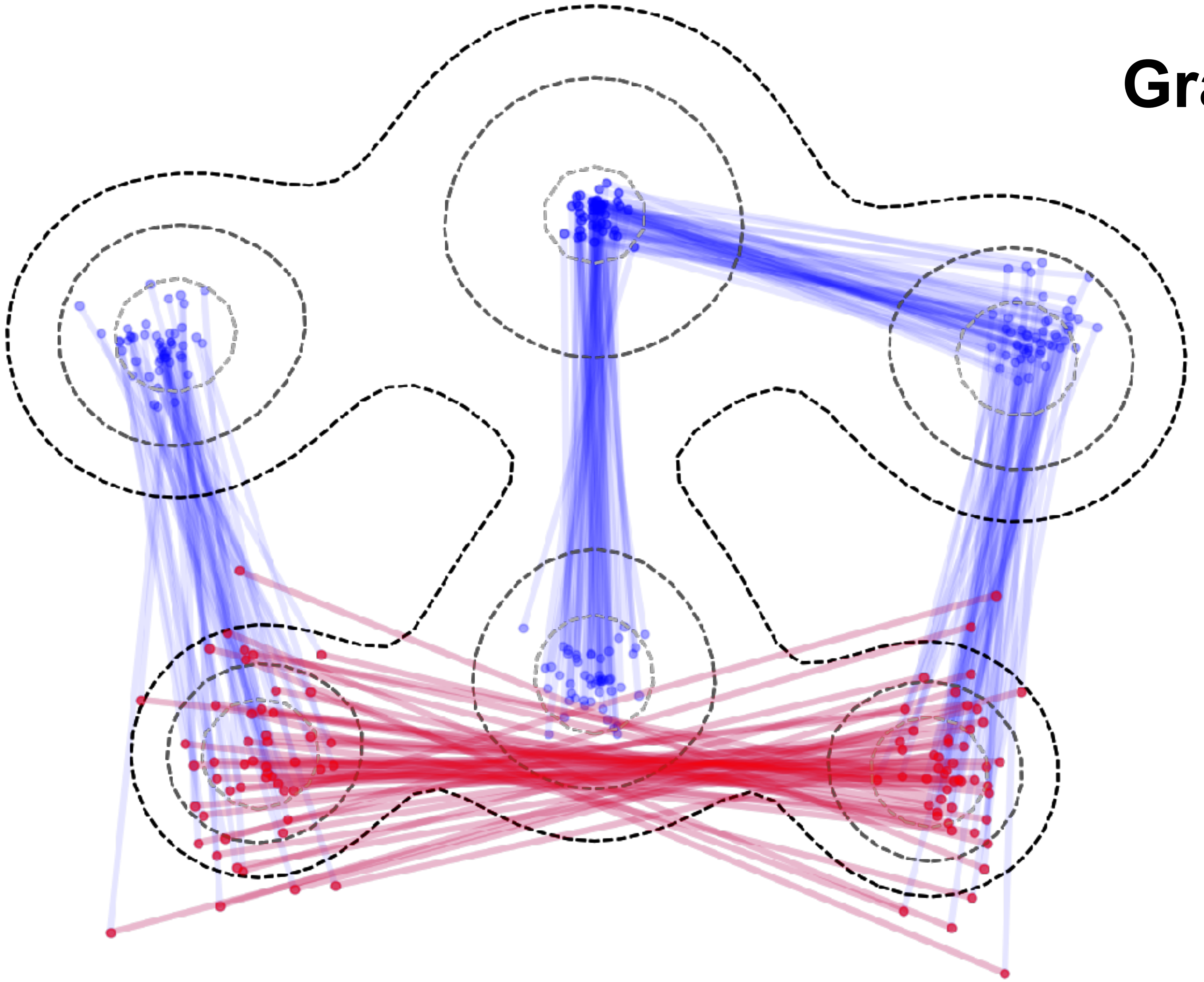
Average amplitude of gradient vector of each Gaussian indicates local flexibility

Statistics on gradient vectors



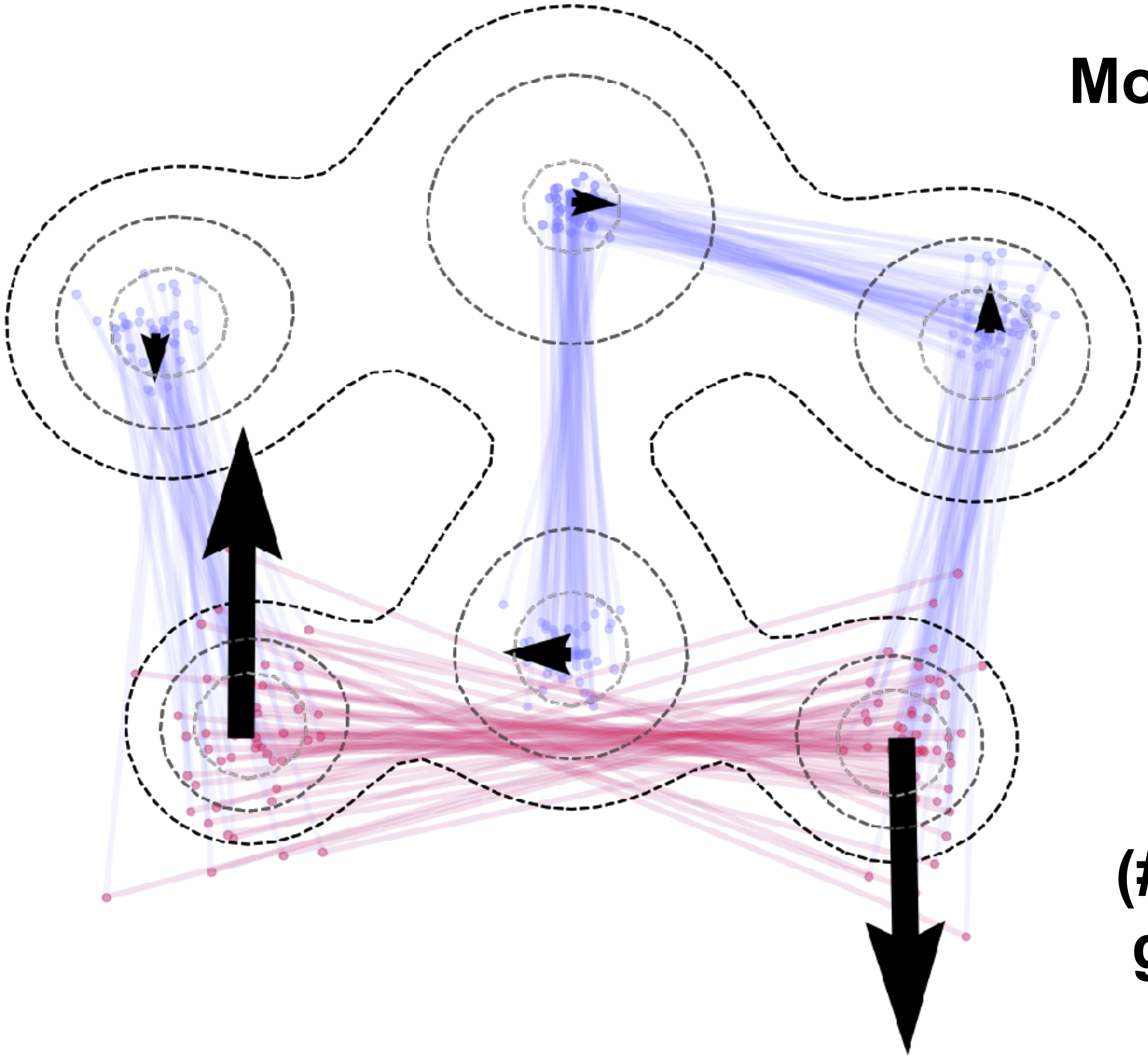
Connect gradient vectors from the same particle

Statistics on gradient vectors



Gradient vectors from each image are correlated

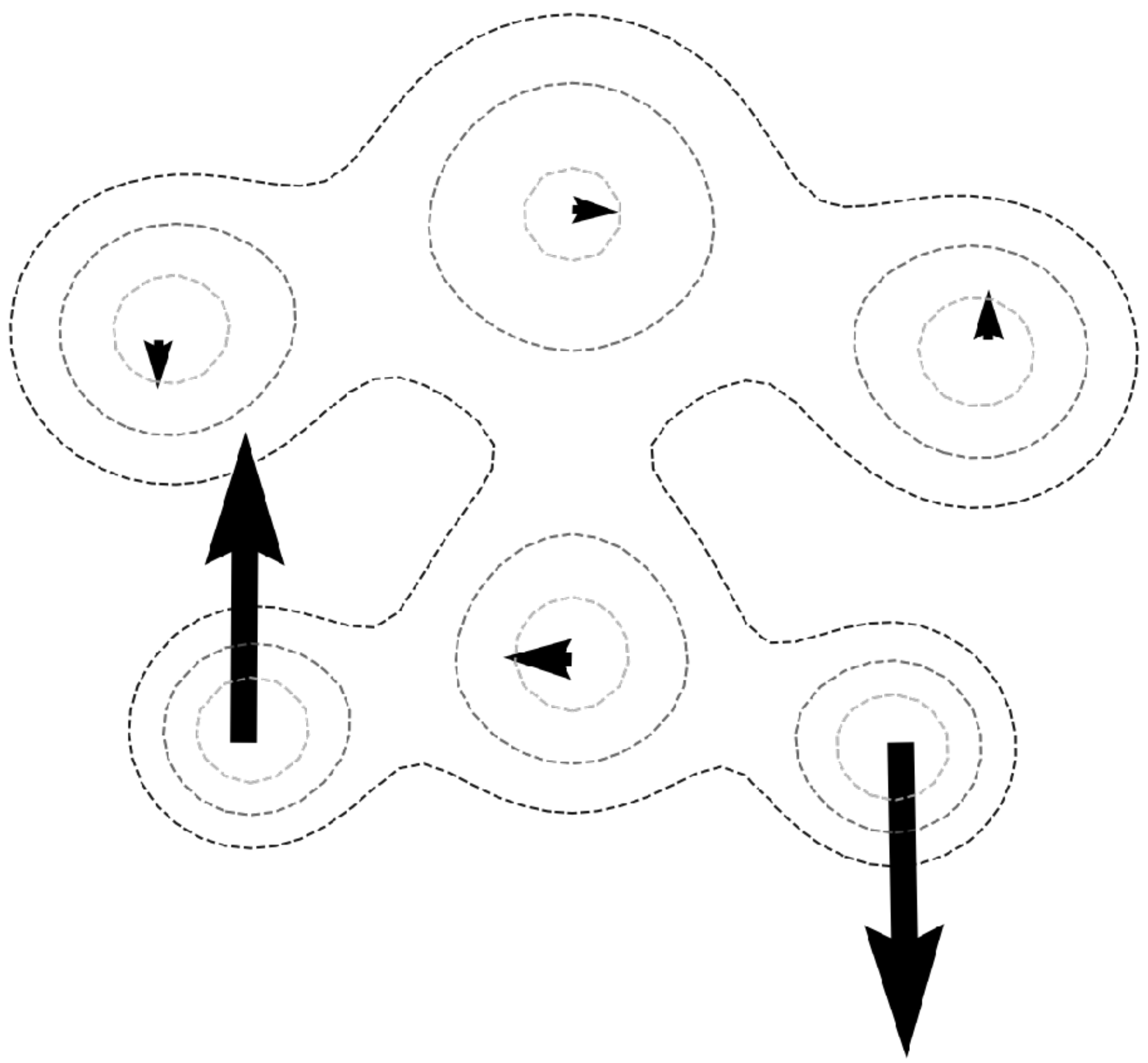
PCA extracts eigen-motion vectors of the system

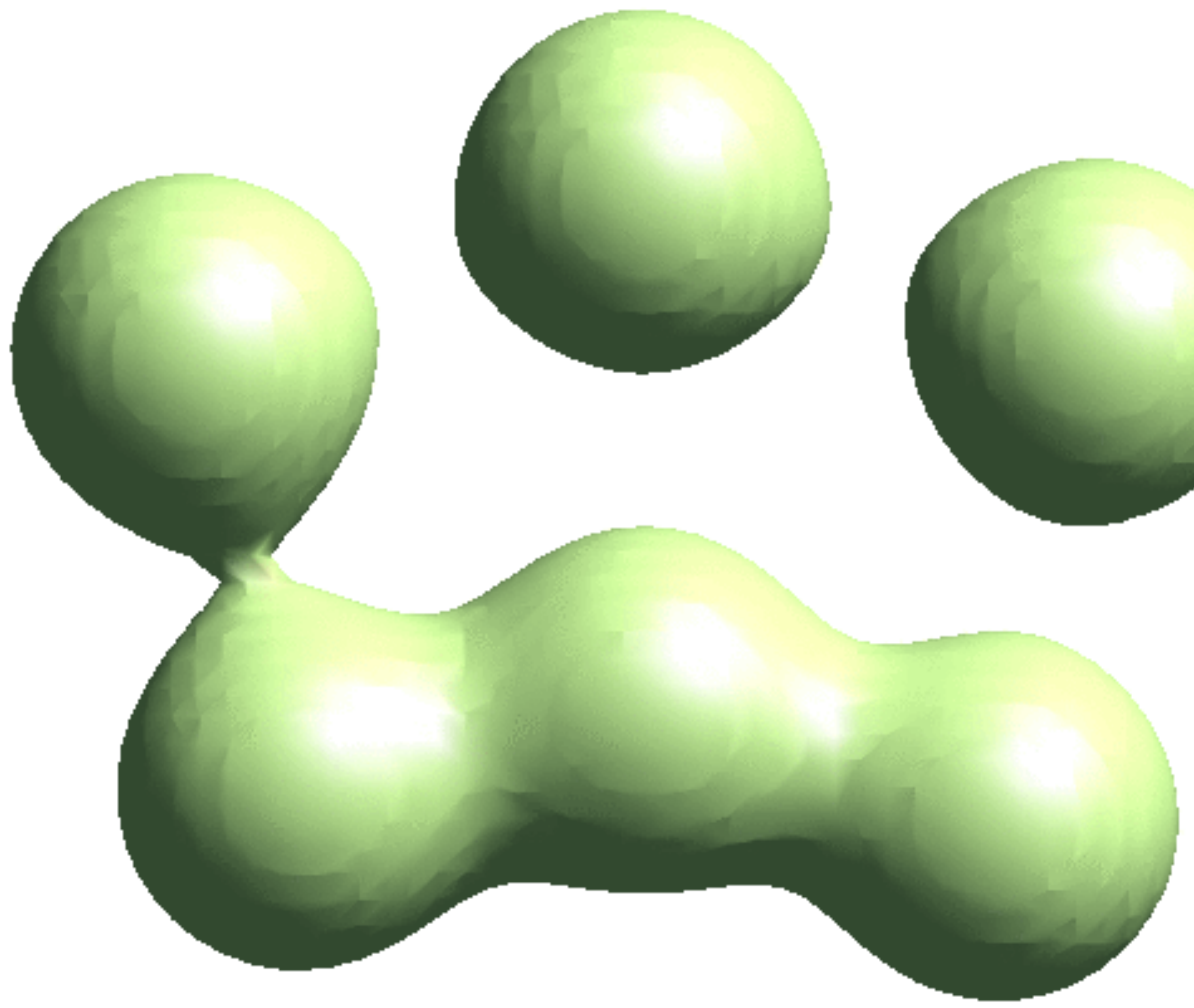


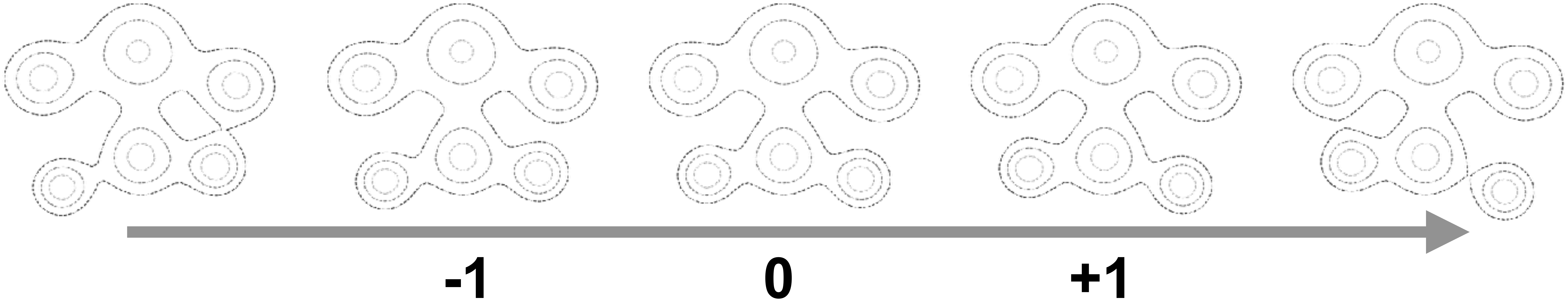
Most particles span on this motion trajectory

This is one vector of length (#Gaussian x 3), which shows a global motion of all Gaussians

Eigen-motion trajectory







Map each particle on the motion trajectory

$$map_i(\mathbf{x}) = \sum_{j=0}^{n} A_j e^{-\frac{\mathbf{x}-(\mathbf{p}_j + c_i \mathbf{v}_j)}{\sigma_j^2}}$$

Particle conformation

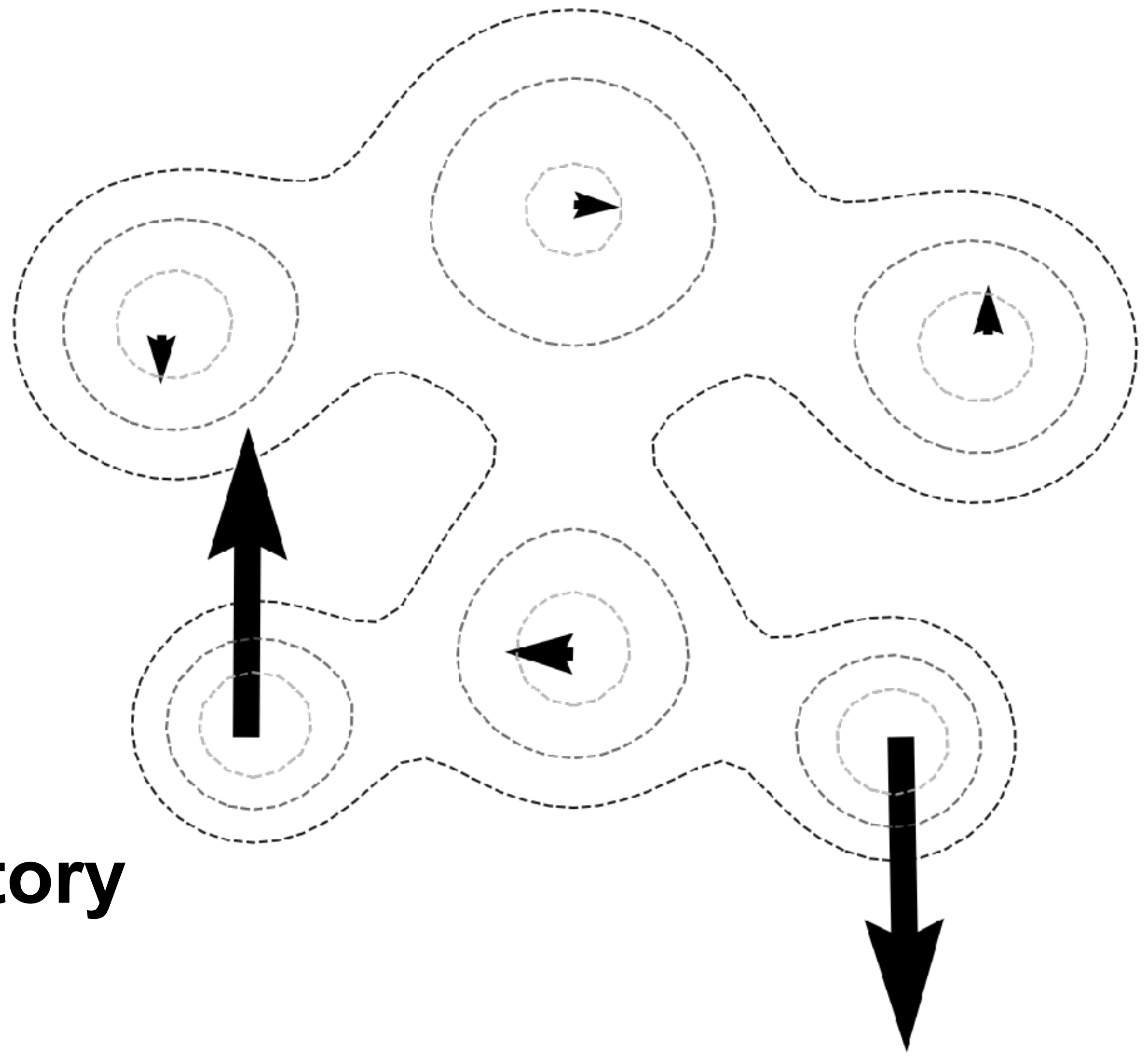
Motion vector

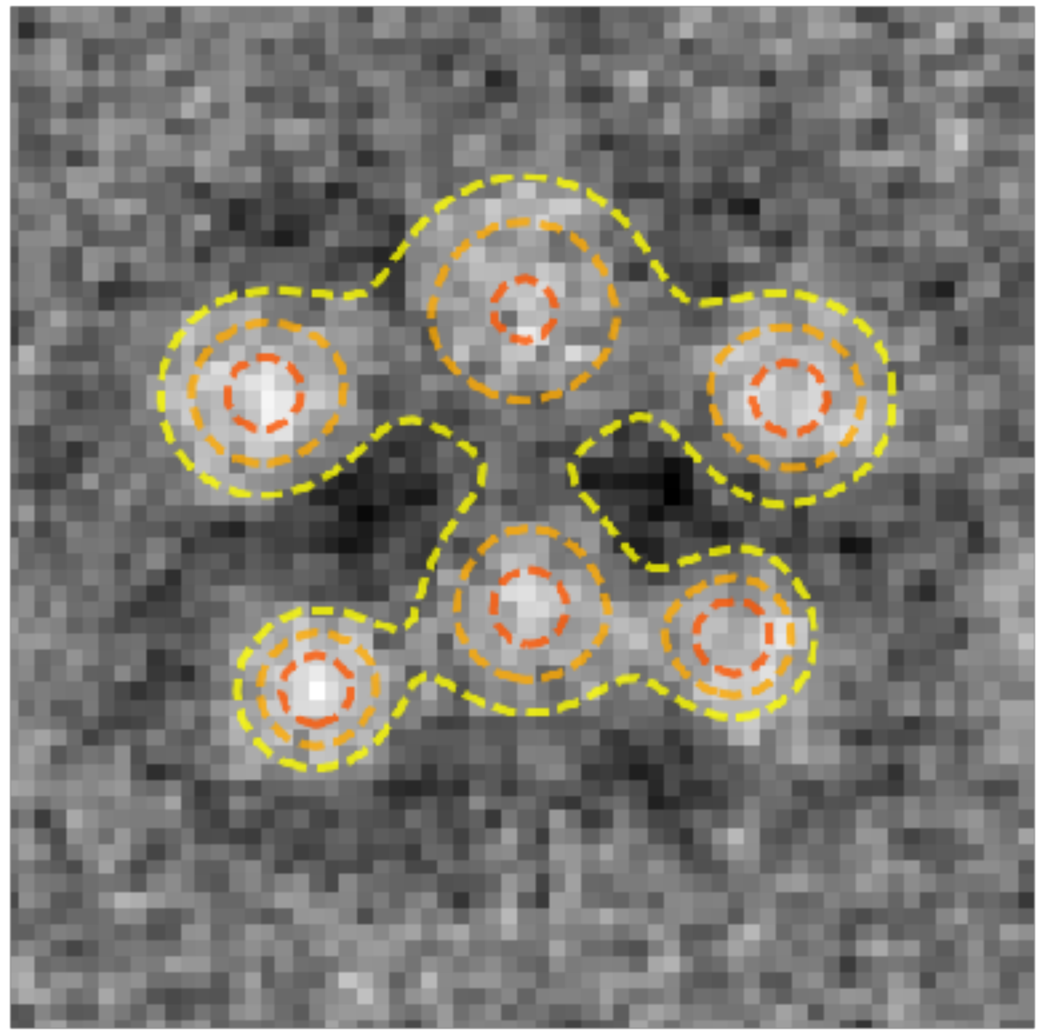
$map_i(\mathbf{x})$: Density map corresponding to the ith particle

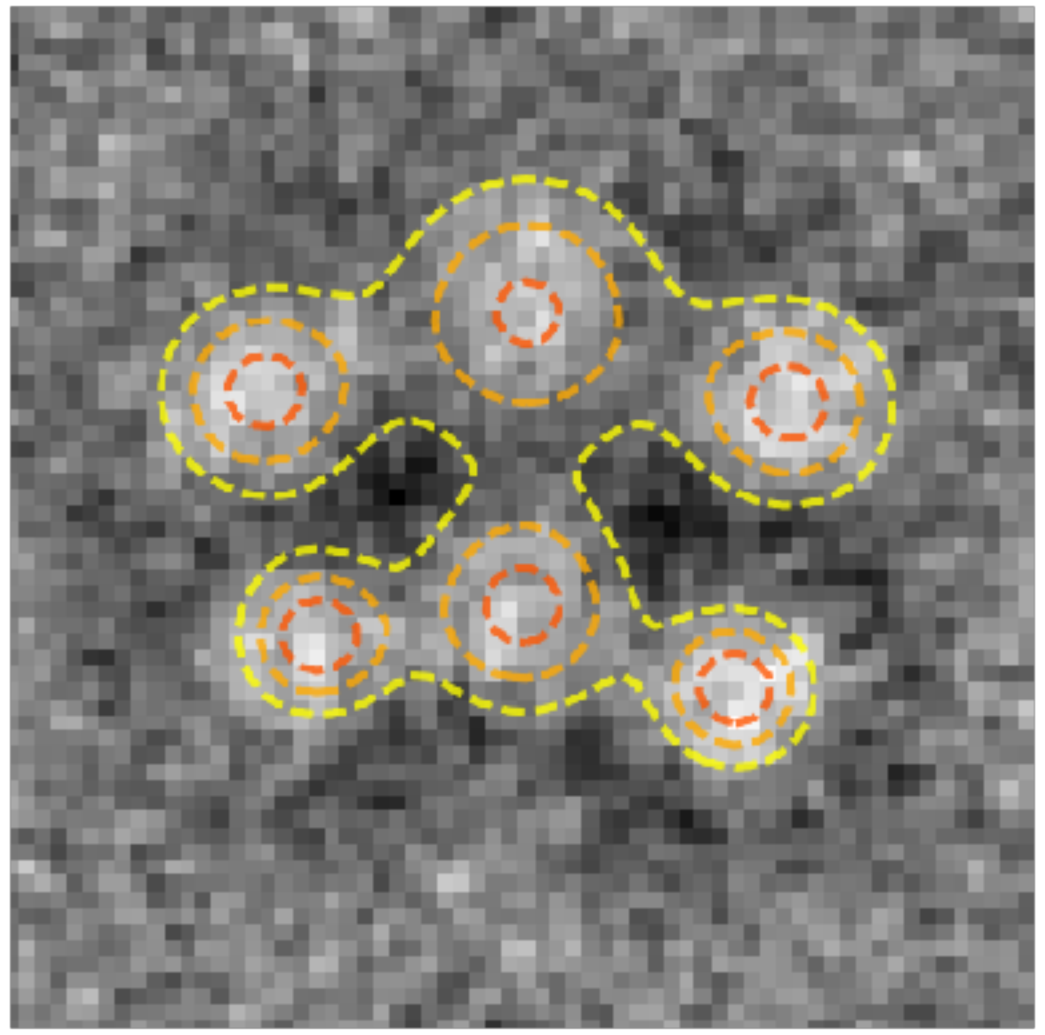
$A_j, \mathbf{p}_j, \sigma_j$: Amplitude, position, sigma of the jth Gaussian

$\mathbf{v}_j$: Eigen-motion vector of the jth Gaussian

c_i : Conformation position of the ith particle on the motion trajectory

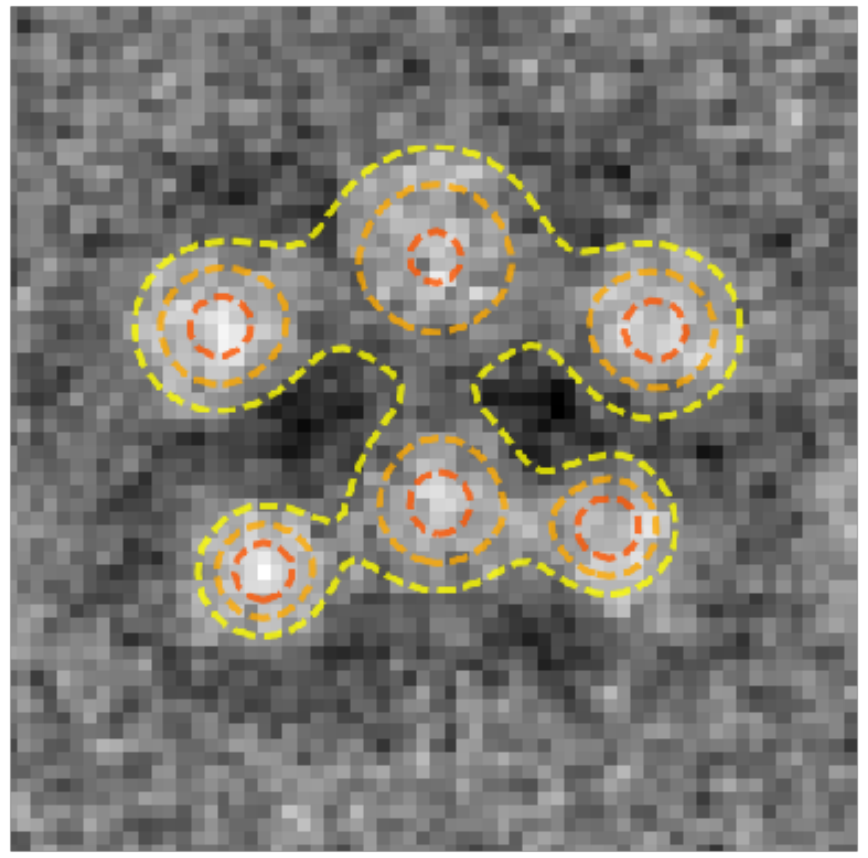


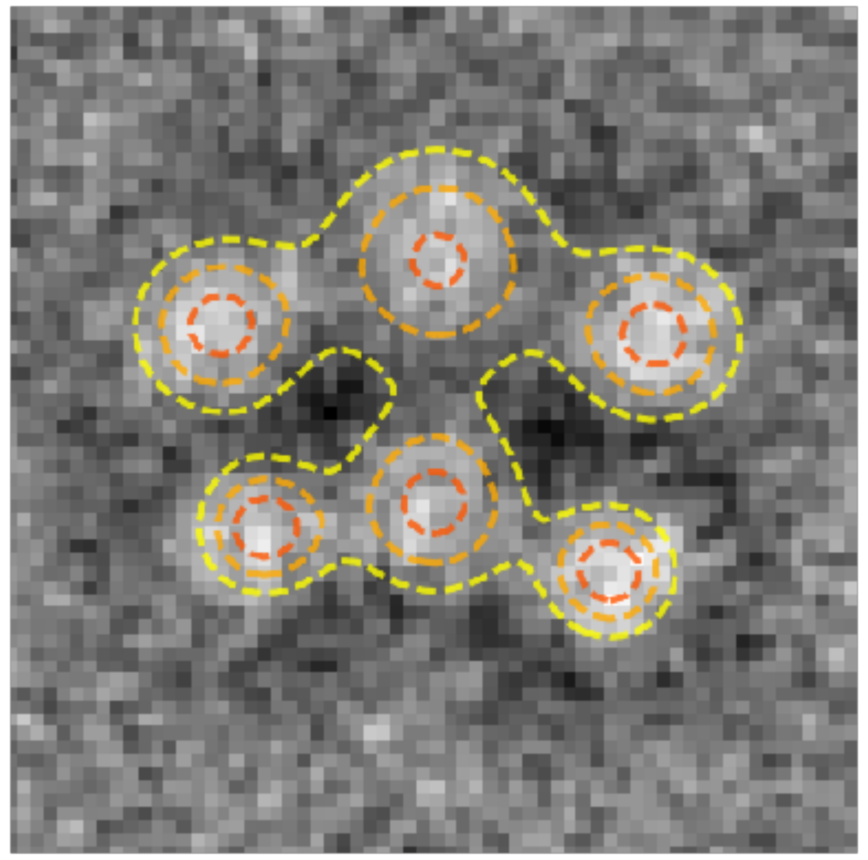


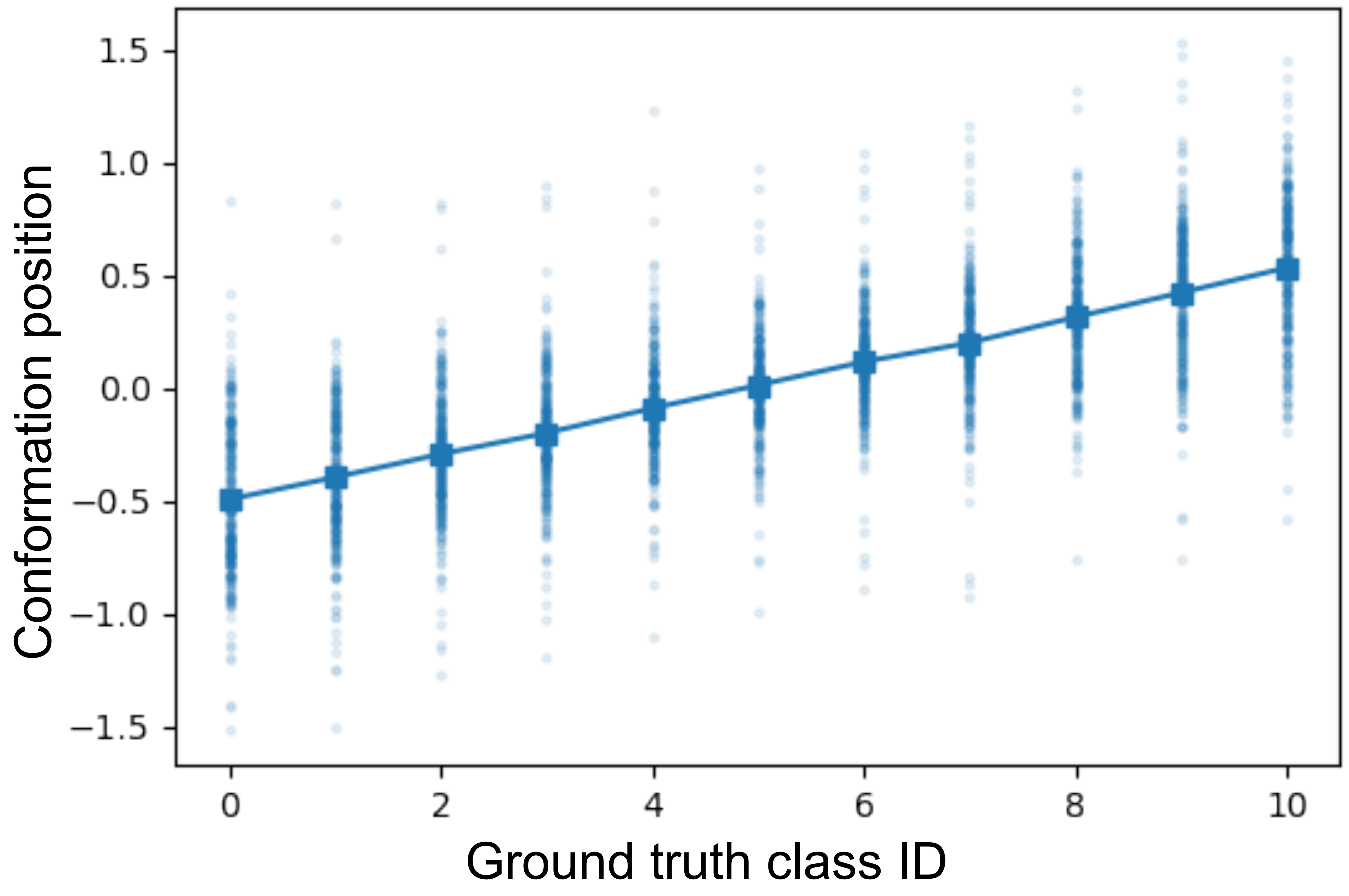


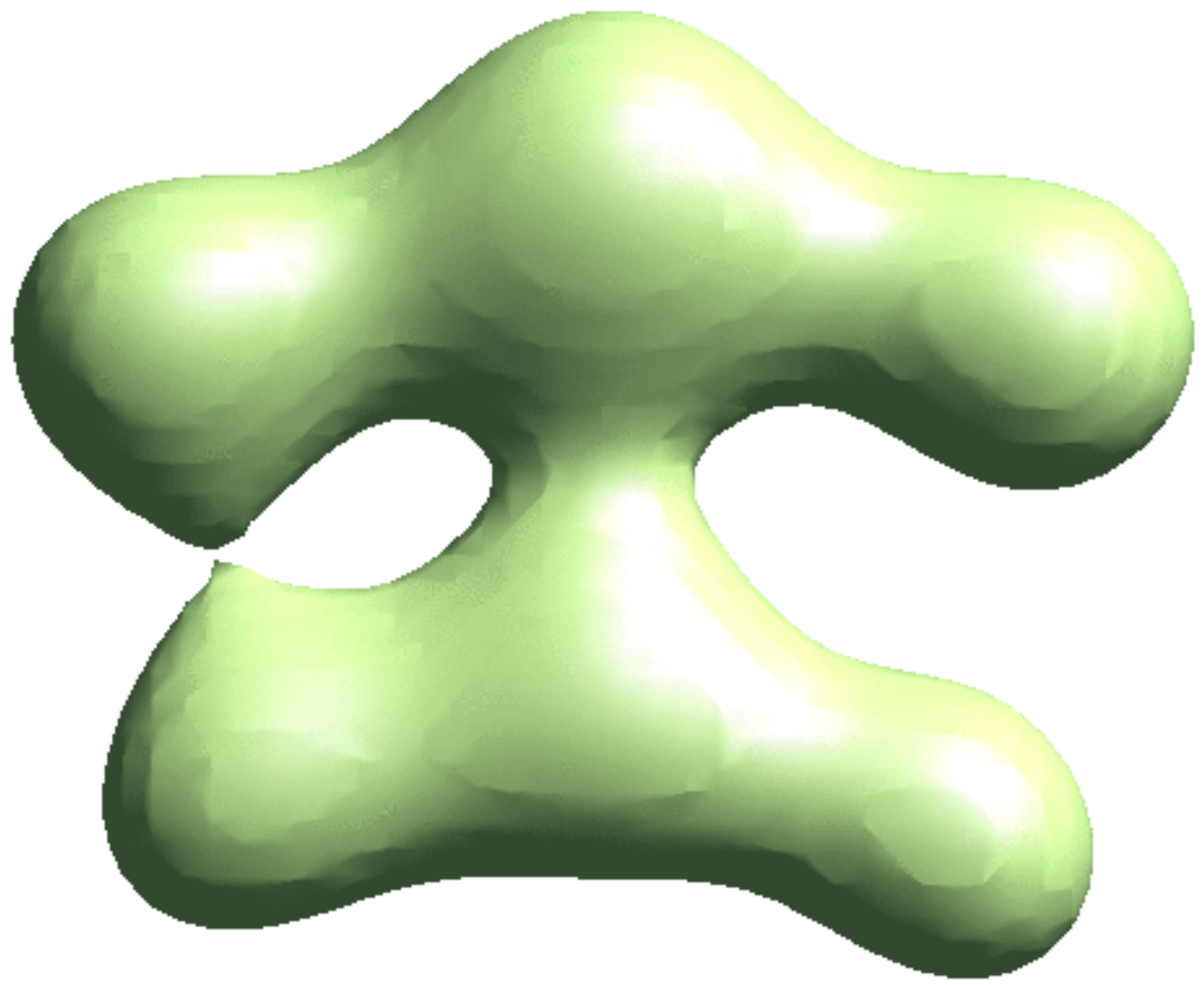
For each particle, optimize c_i with gradient descent

Reconstruct 3D map with particles of similar conformations





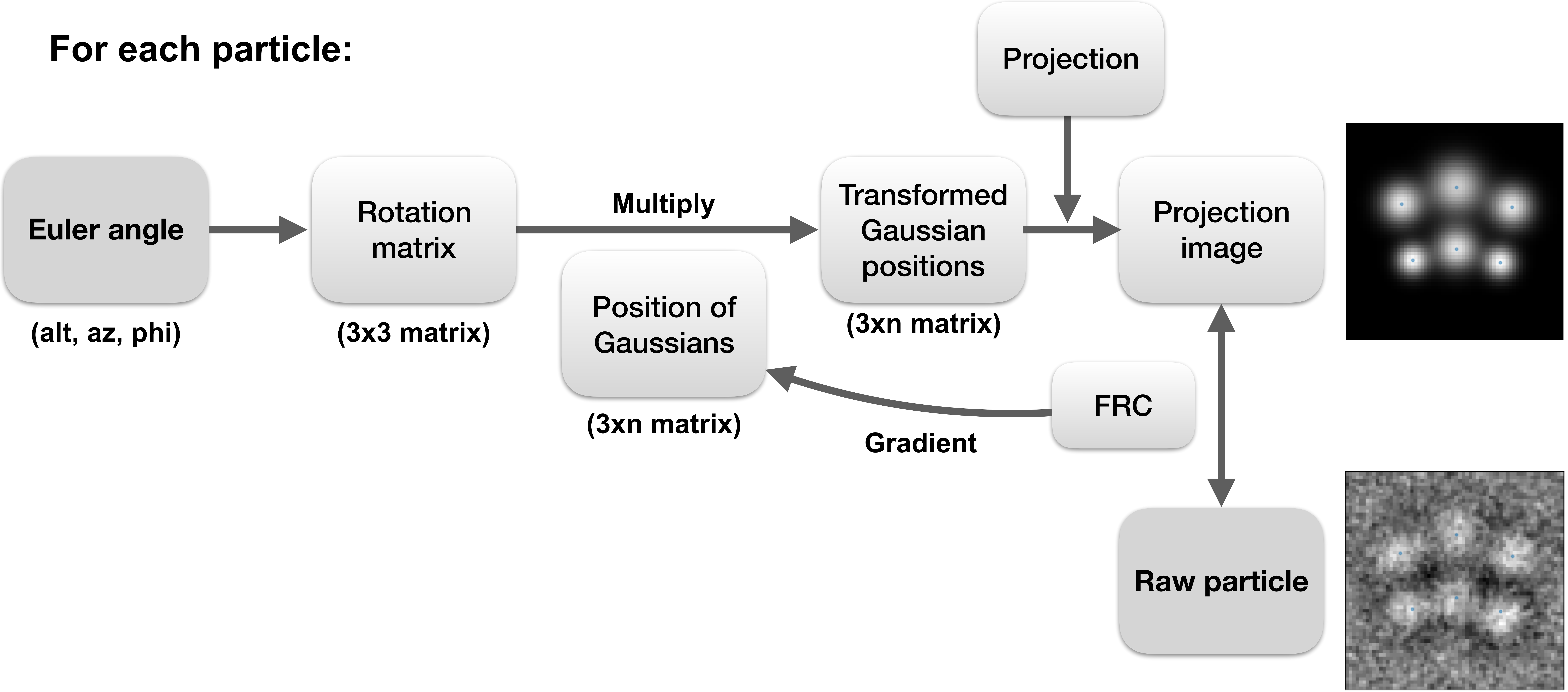




planar motion in this simulated dataset (invisible at z direction) exaggerates top view error…

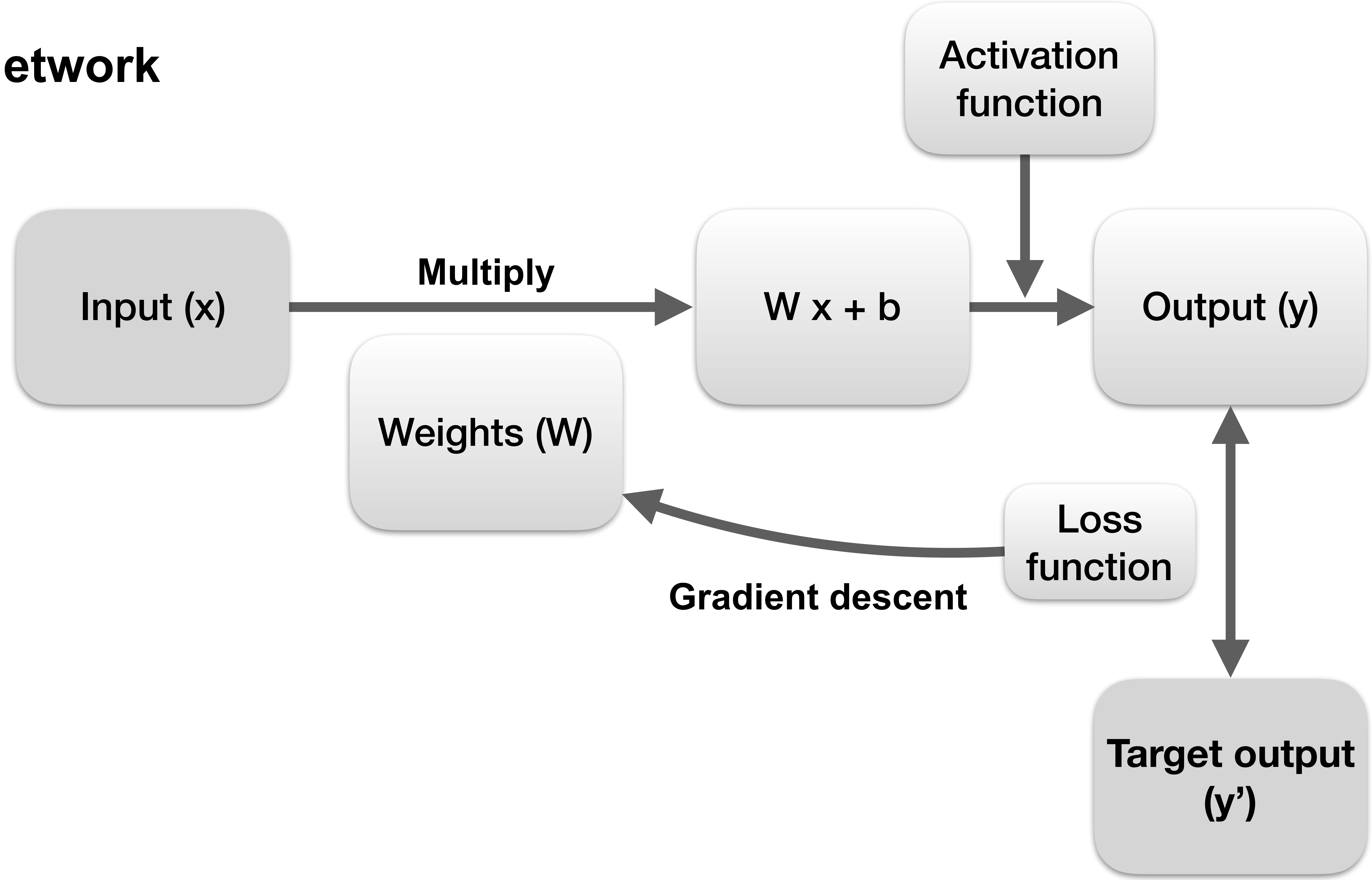
Relationship with neural network

For each particle:



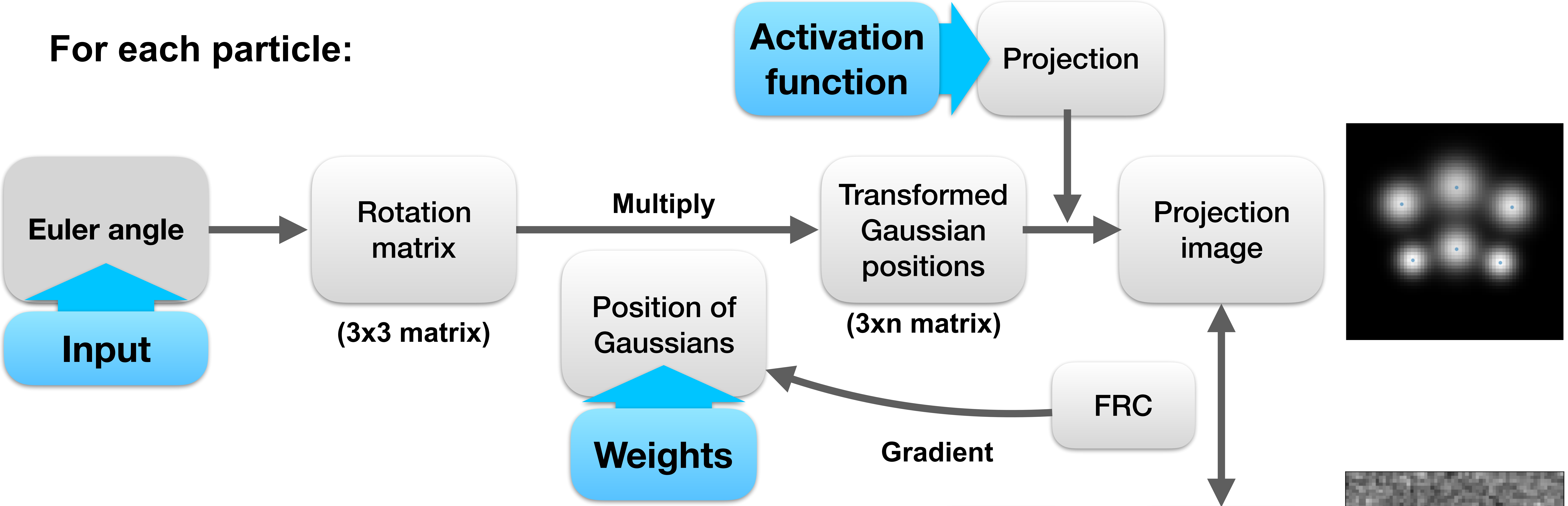
Relationship with neural network

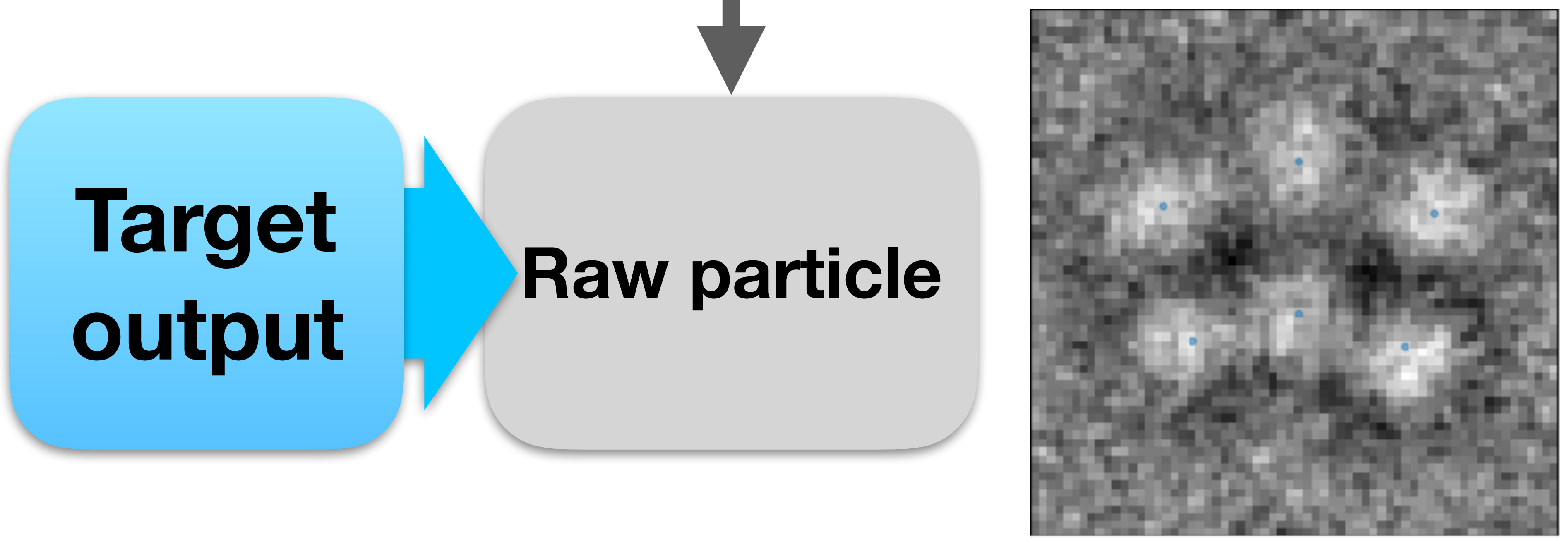
One layer neural network



Relationship with neural network

For each particle:

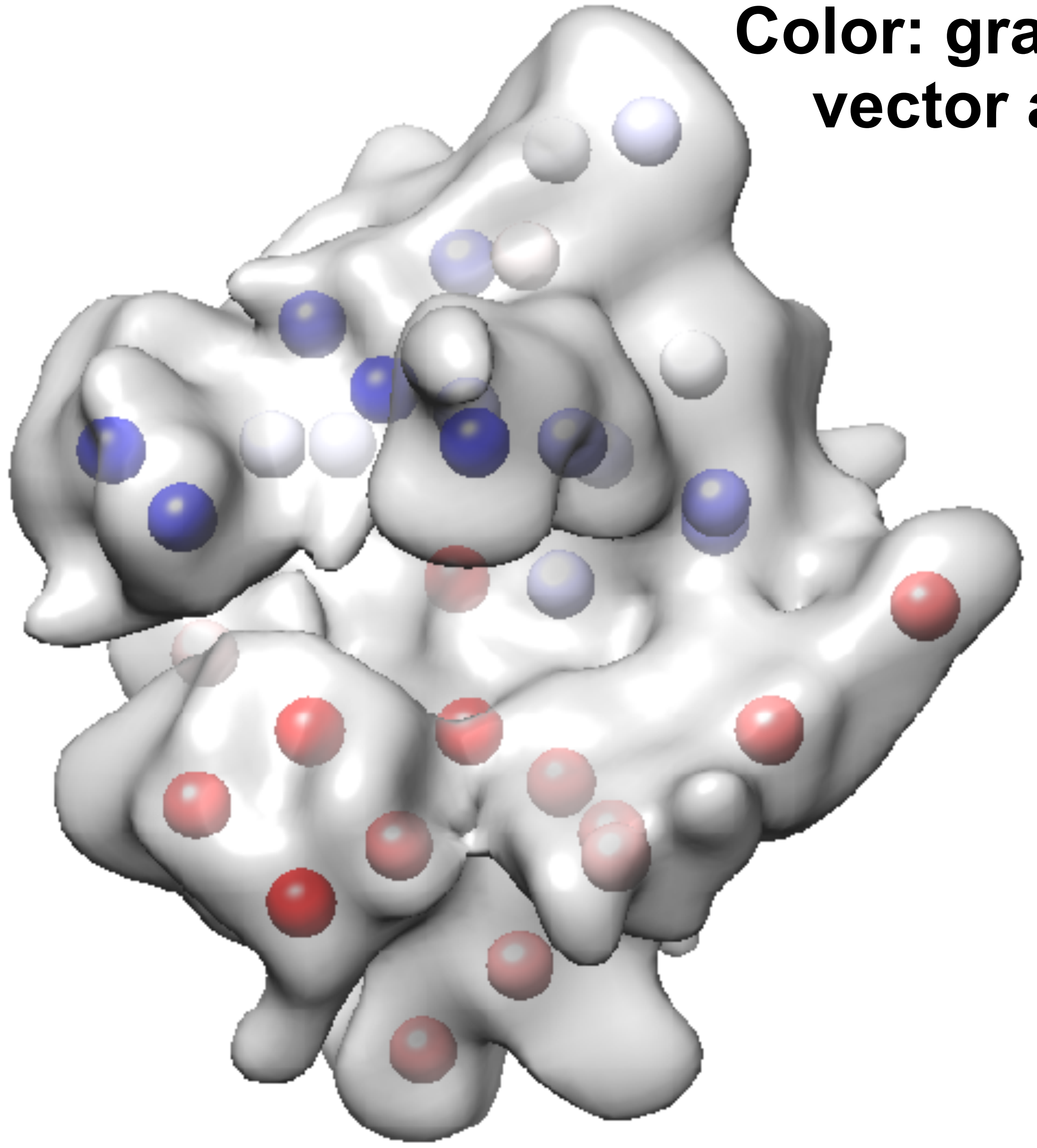




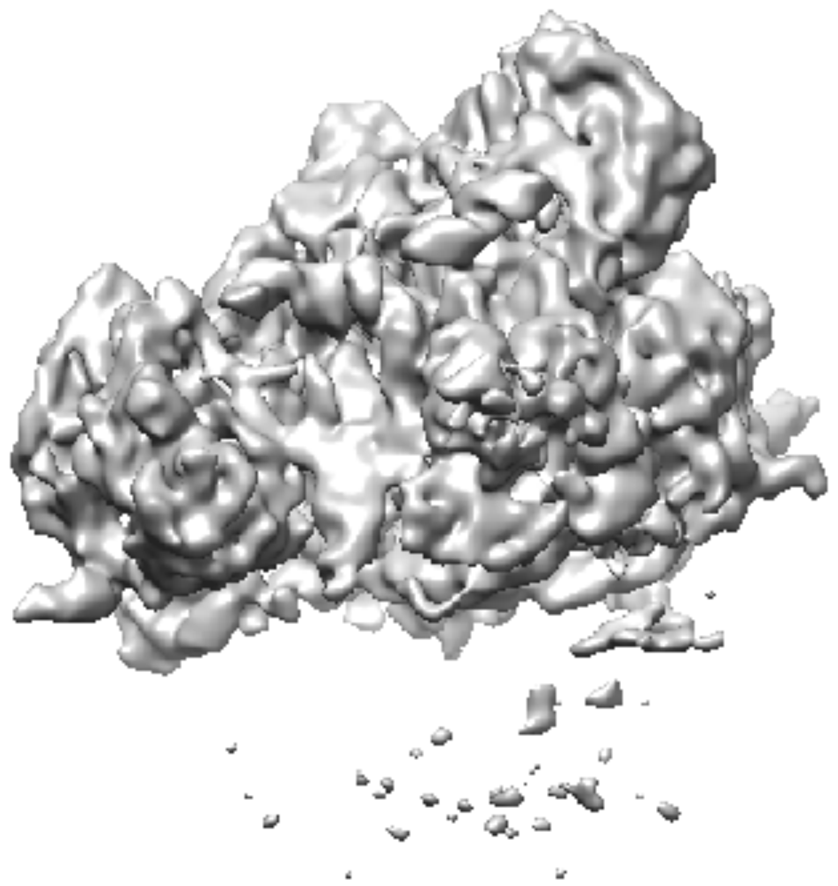
- Implement under deep learning framework
- Efficient GPU utilization
- Symbolic gradient calculation

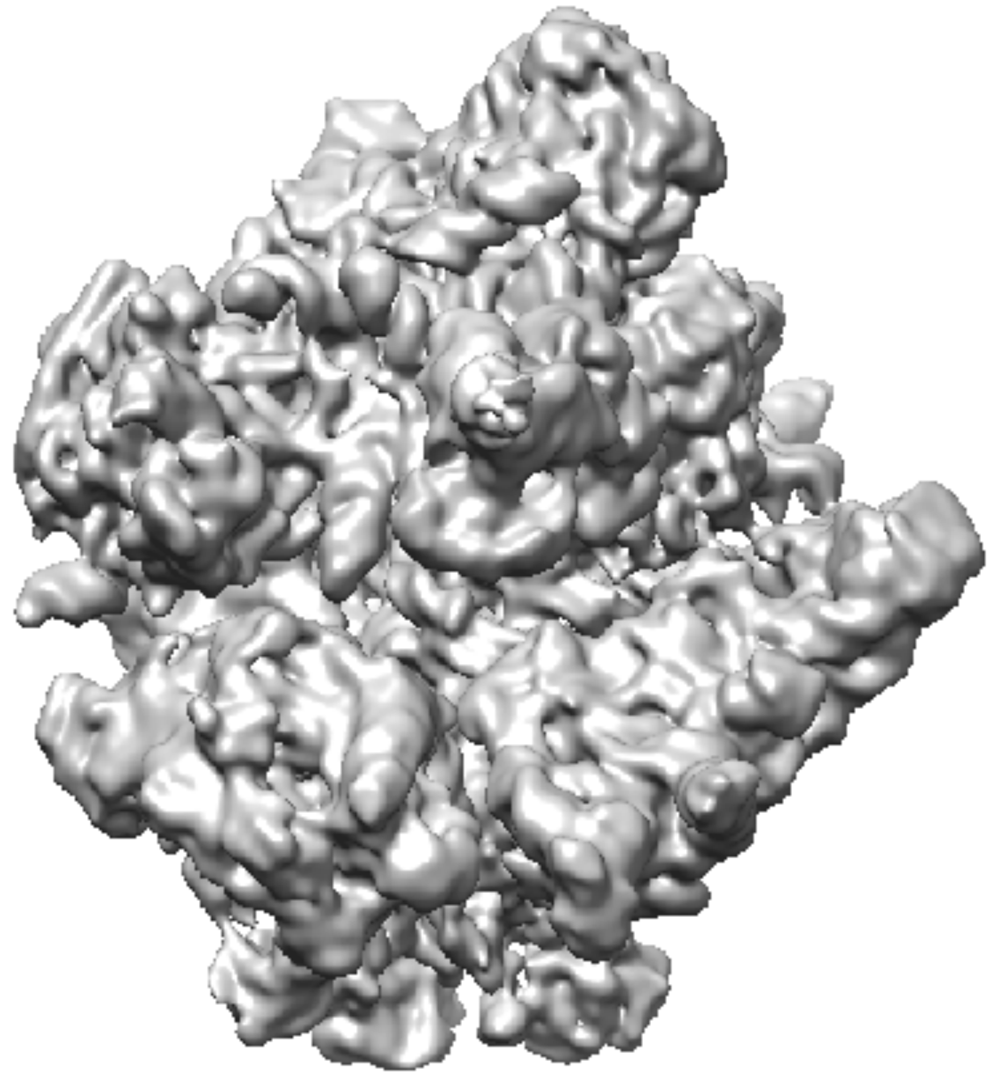
Performance on real examples

Ribosome (EMPIAR-10107)



Color: gradient eigen-vector amplitude



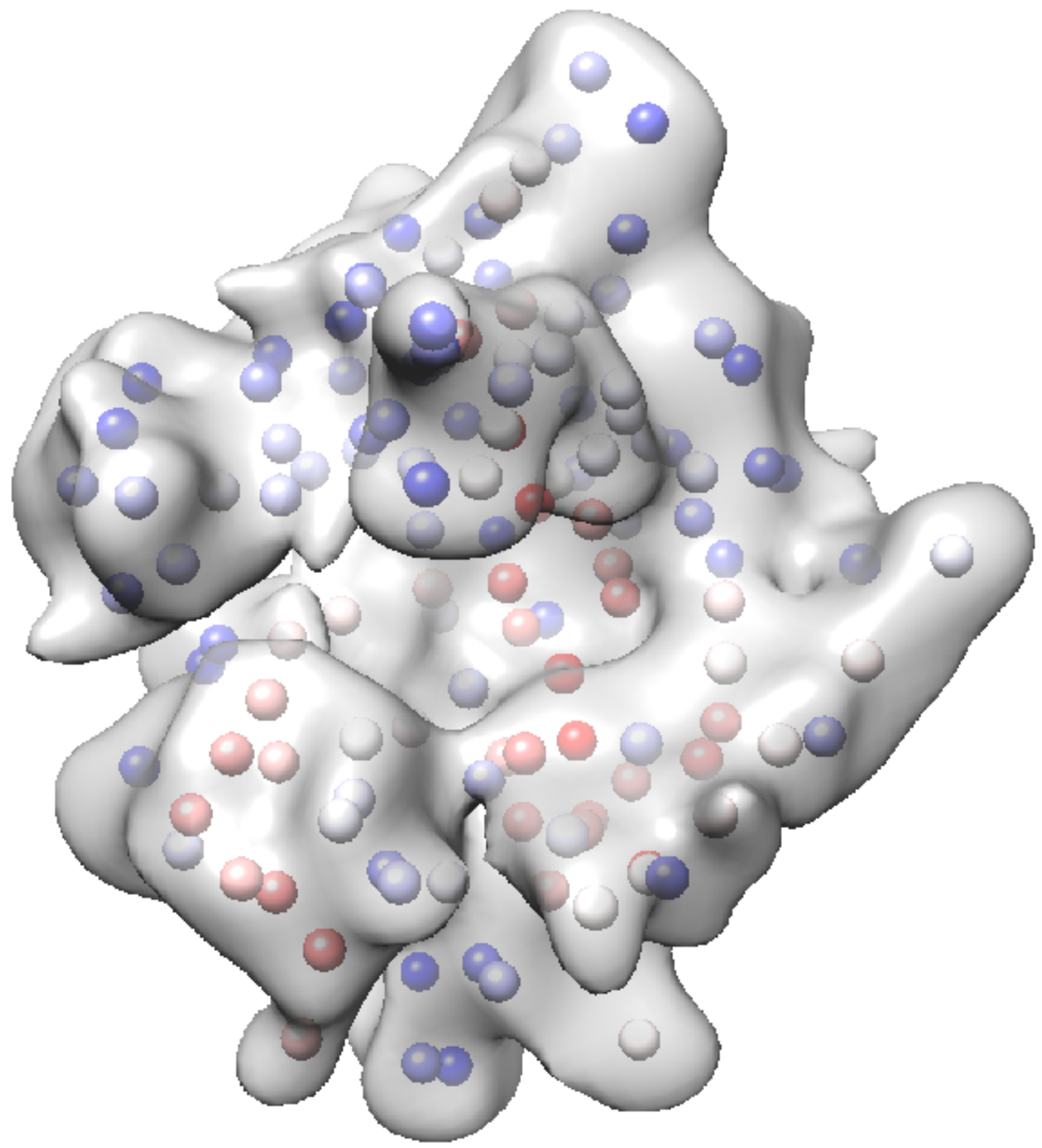


$$Grad_j = \frac{\partial FRC}{\partial A_j}$$

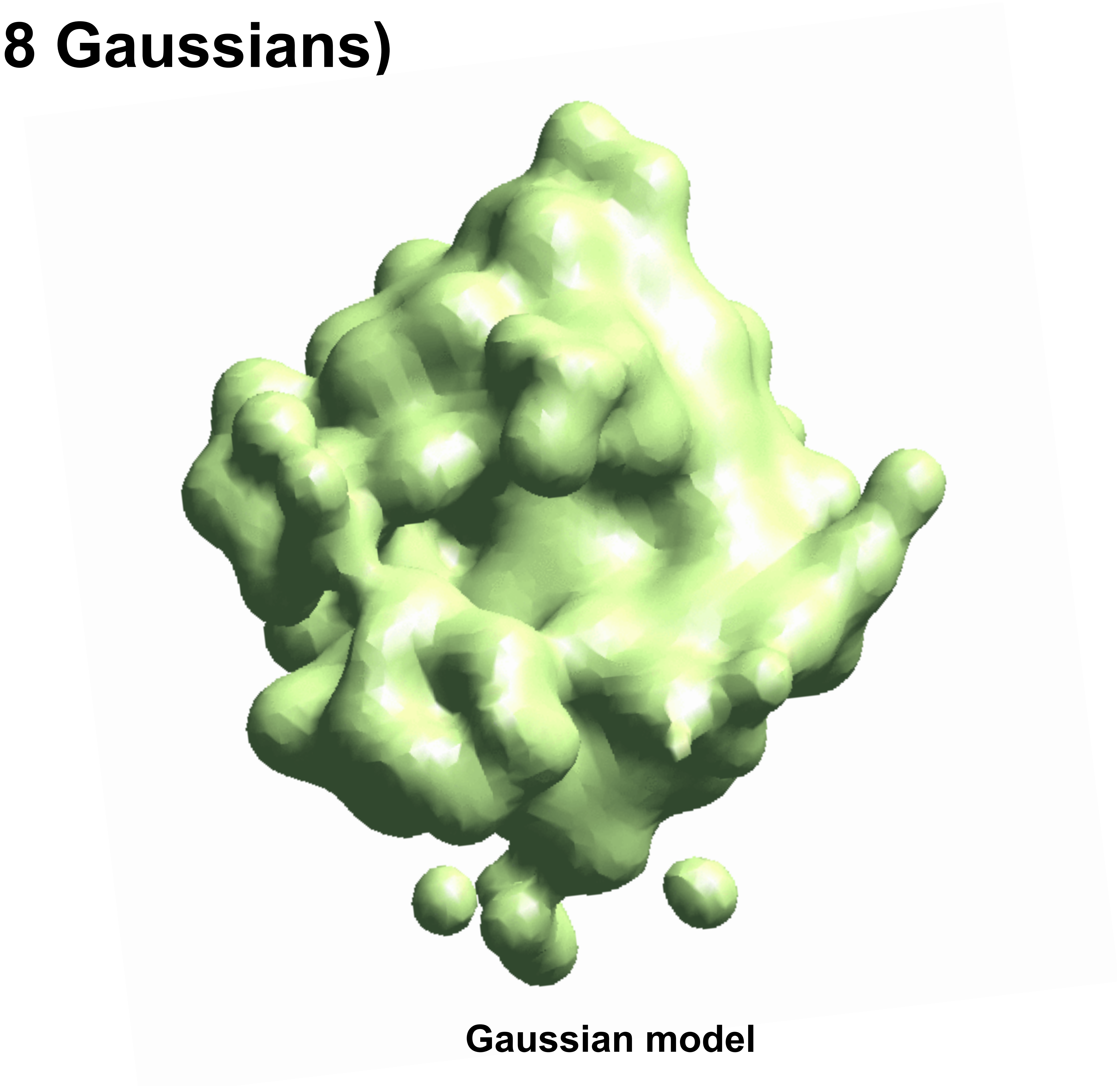
Compositional heterogeneity (50Å, 64 Gaussians): gradient with respect to amplitude of each Gaussian

Ribosome (EMPIAR-10107)

Conformational heterogeneity (40Å, 128 Gaussians)

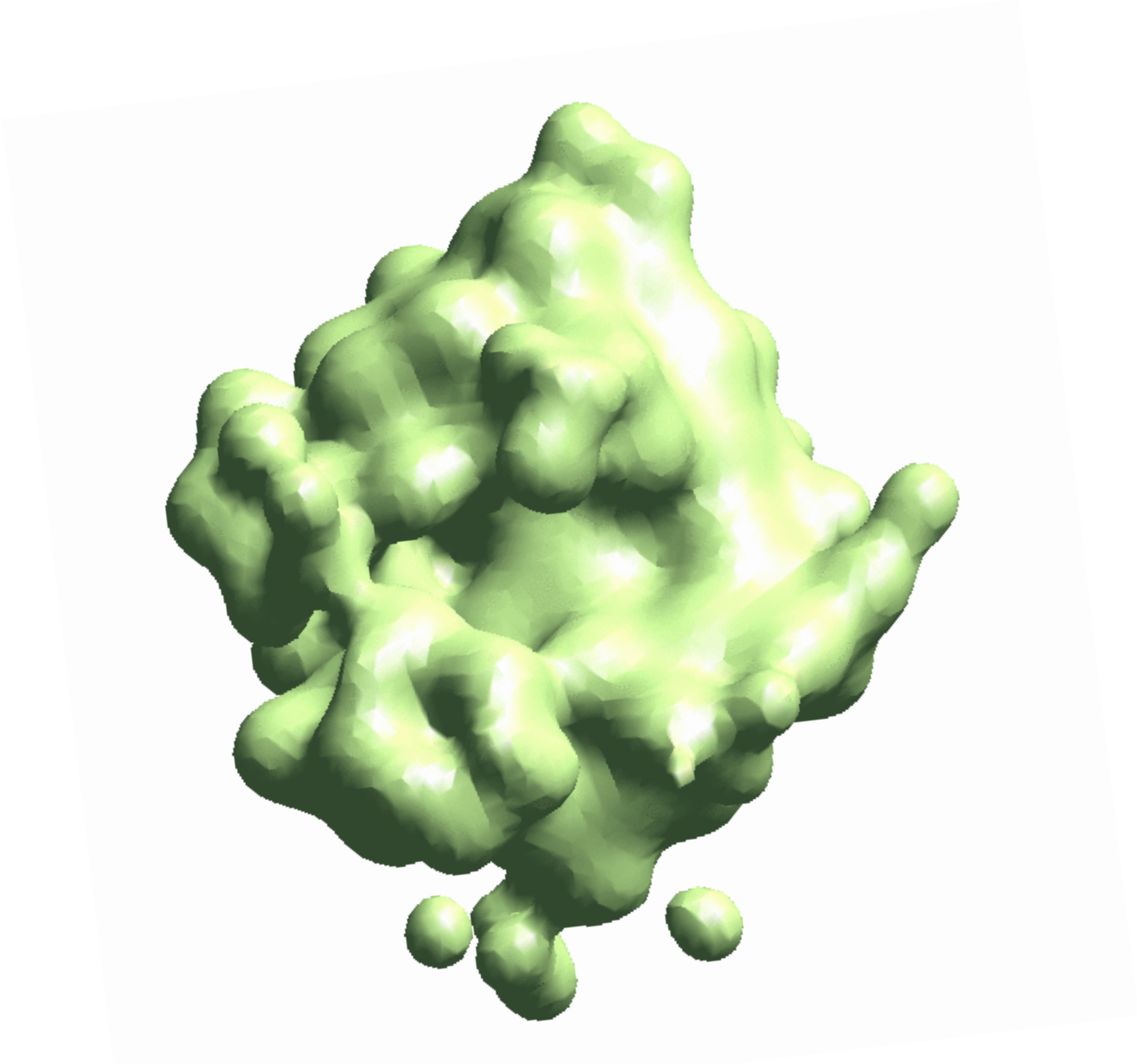


Color: eigen-motion amplitudes

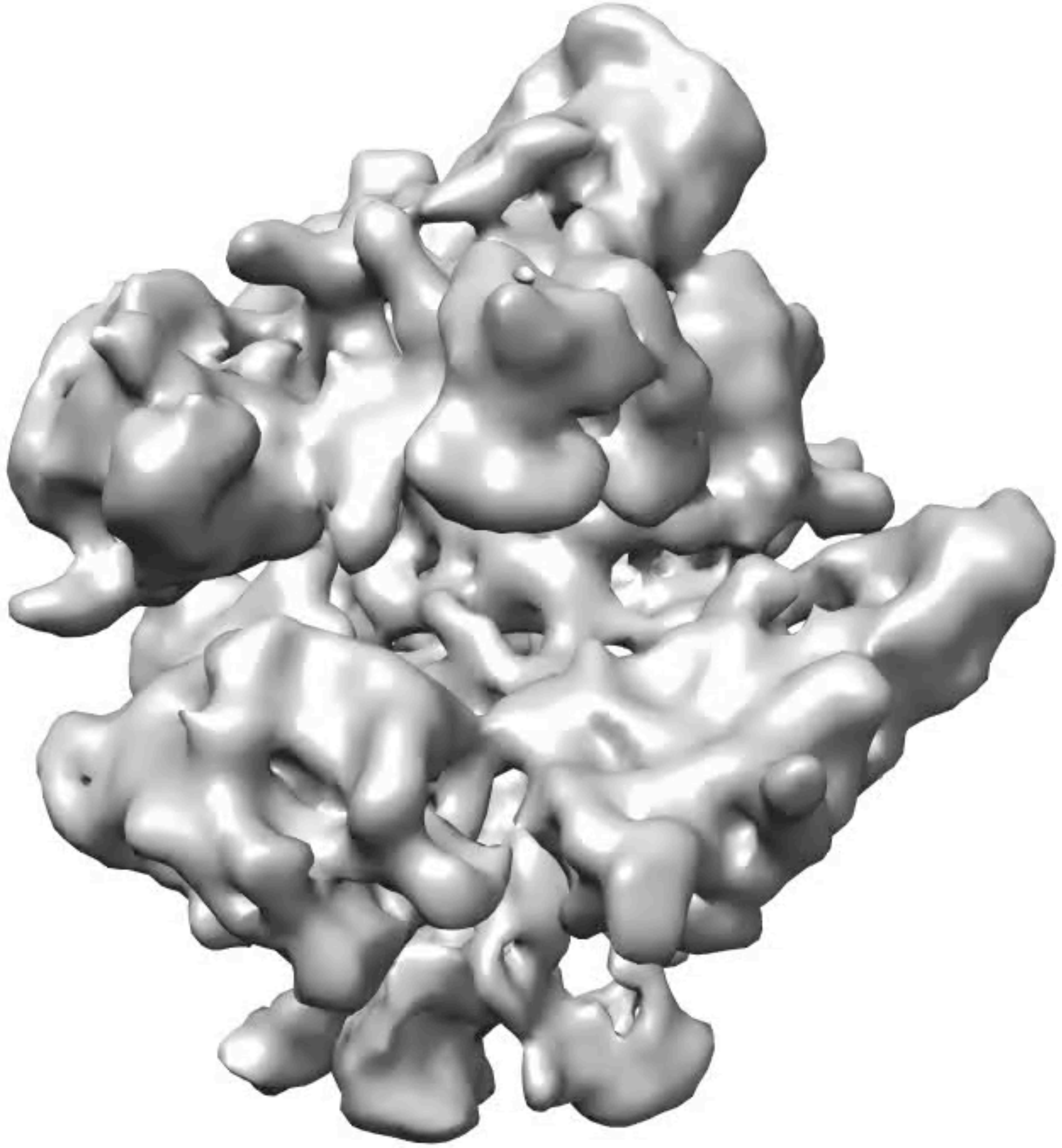


Gaussian model

Global motion

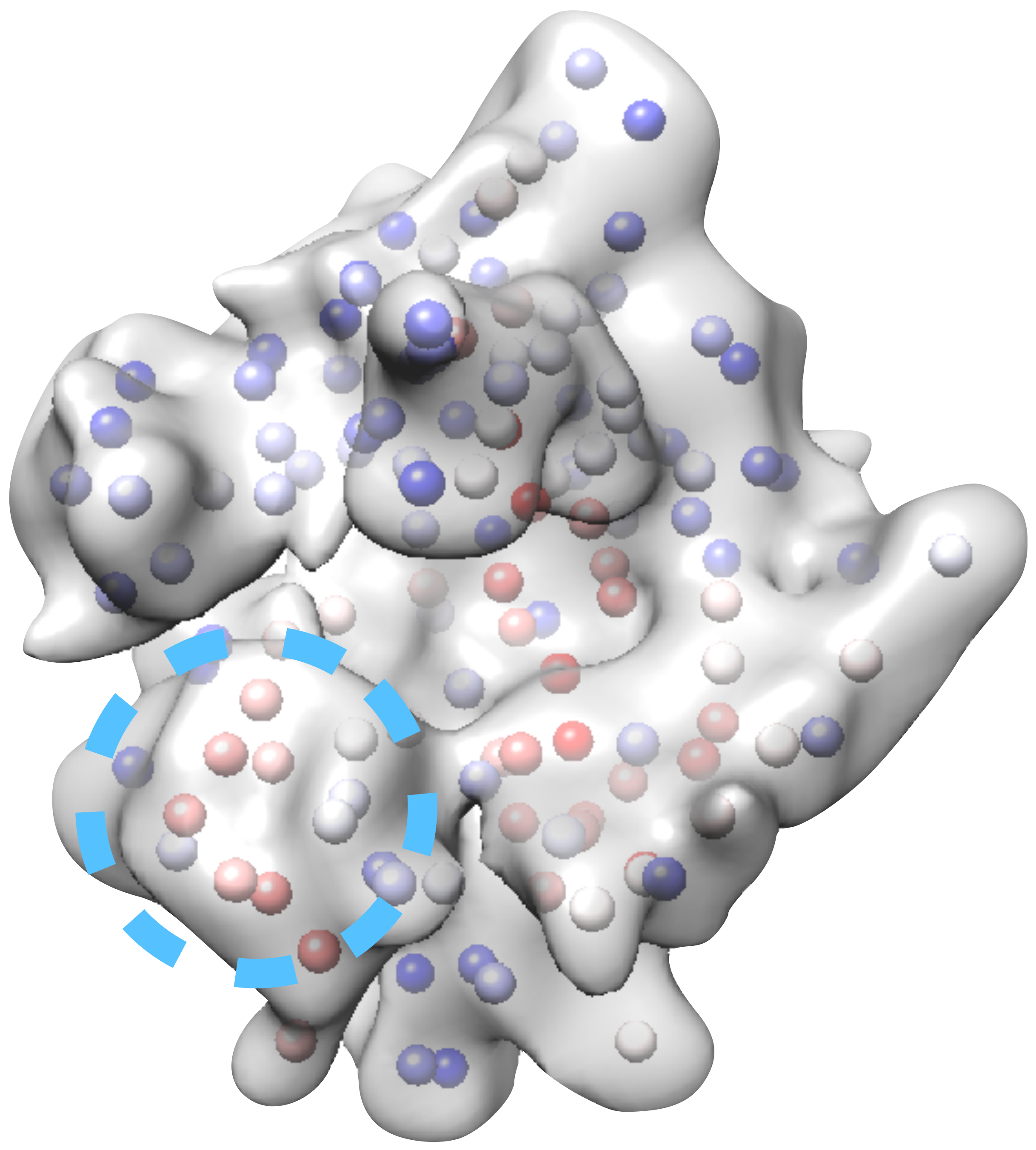


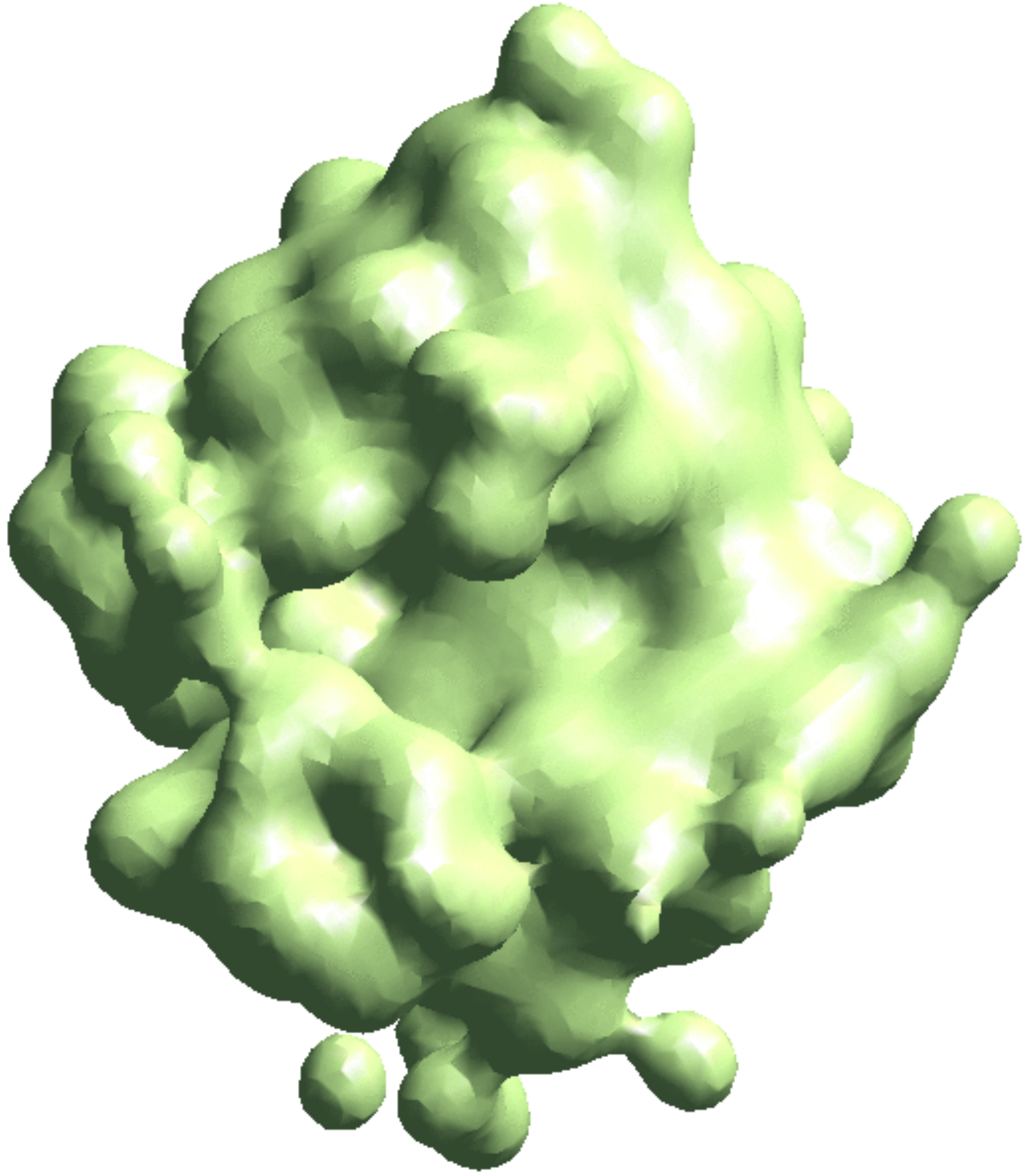
Gaussian model



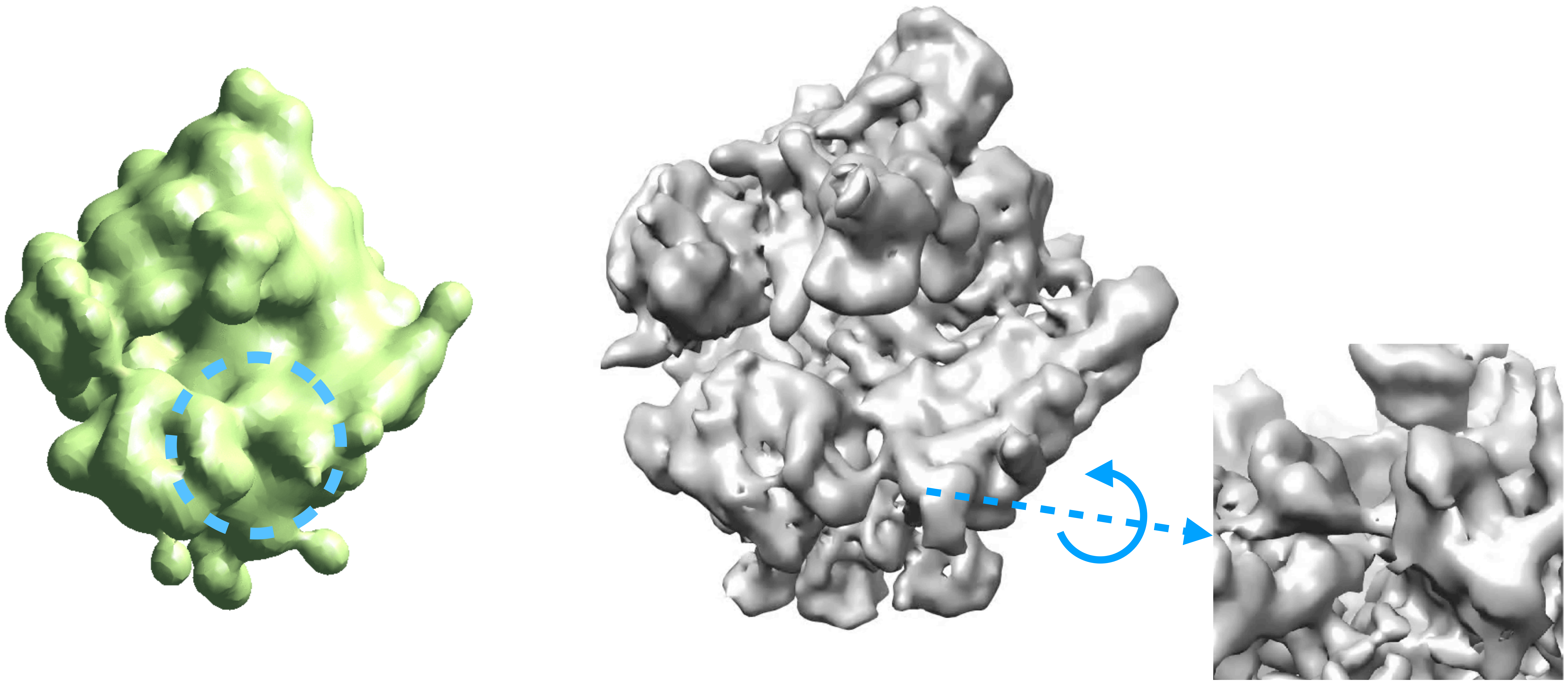
Reconstructed maps from
classified particles

Focus on local regions

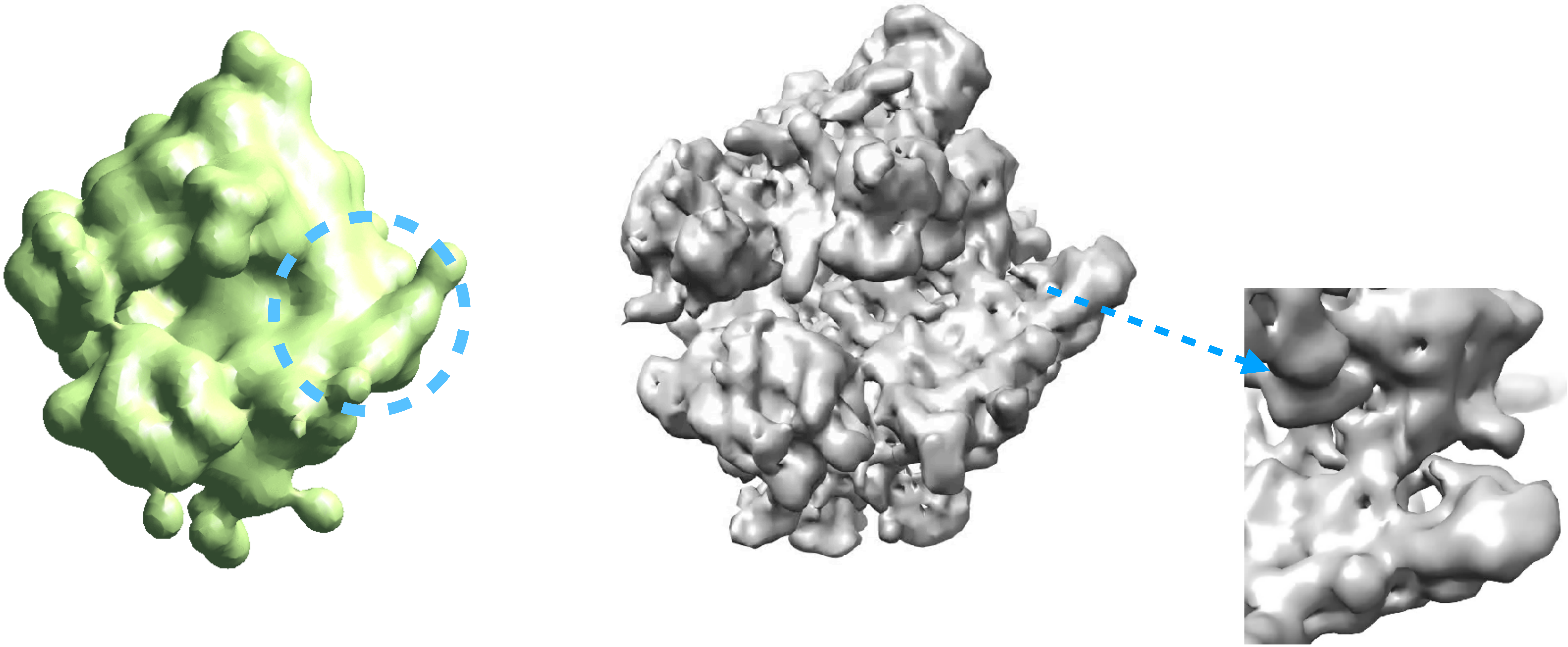




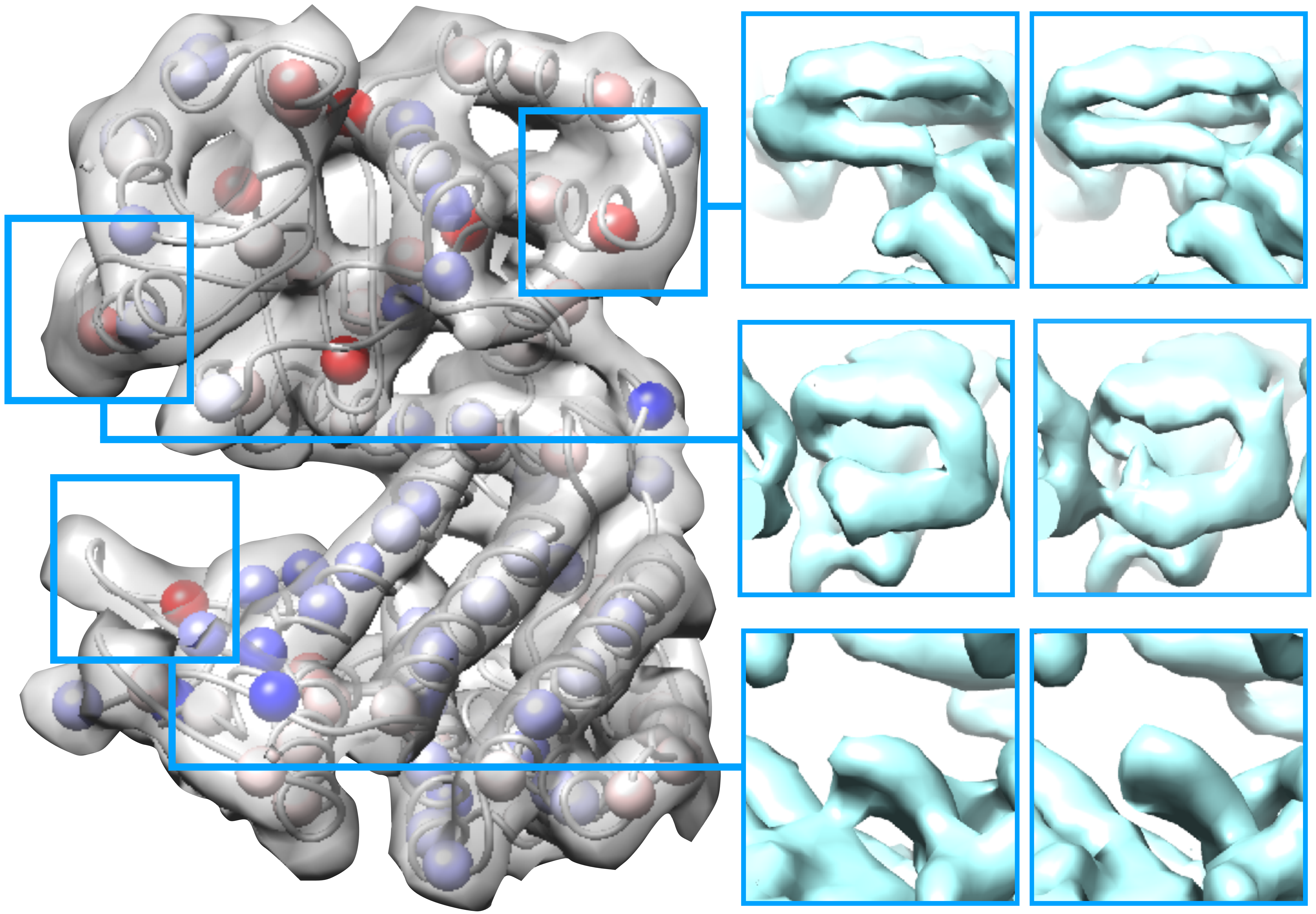
Focus on local regions



Focus on local regions



GroEL



- From Roh 2017, *PNAS*
- Filtered to 7Å
- 1344 Gaussians
- Motion of helices
- Symmetry breaking
- Correlation of conformation between subunits

Advantages

VS multi-model refinement:

- Deterministic
- Handles continuous motion

VS image based manifold mapping:

- Lower requirement for dataset
- Simple, expandable framework
- Solves global and local motion

Limitations

- Requires determined orientations from single model refinement
- Linear and short motion trajectory
- Low resolution due to limited GPU memory

Advantages

VS multi-model refinement:

- Deterministic
- Handles continuous motion

VS image based manifold mapping:

- Lower requirement for dataset
- Simple, expandable framework
- Solves global and local motion

Limitations
and future directions

- Requires determined orientations from single model refinement
  - Iteratively optimize conformation and orientation
- Linear and short motion trajectory
  - Replace PCA with stacked autoencoder for trajectory calculation
- Low resolution due to limited GPU memory
  - Better hardware and software platforms

Availability

- EMAN2/examples
  - *build_ali_lst.py*
    build list file with alignment information from existing refinement
  - *gmm_heterog.py* or *gmm_heterog_tensorflow.py*
    main program (theano or tensorflow implementation)
- Tutorial coming soon…

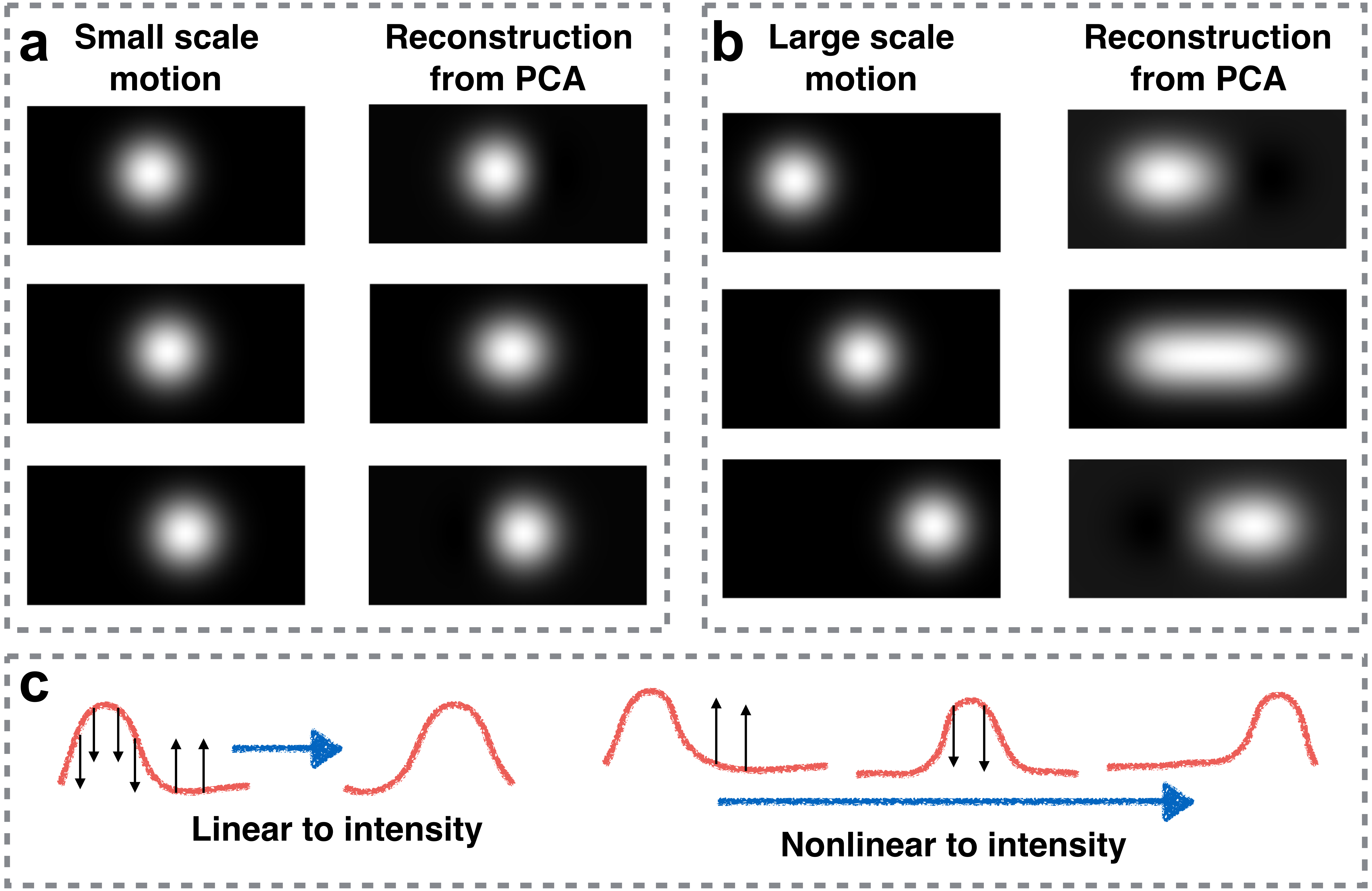
Acknowledgement

- PIs:
  - Steven Ludtke, *Baylor College of Medicine*
- NIH grants:
  - R01GM080139

Special thanks to Hui Ye for looking after my cat…

Thank you

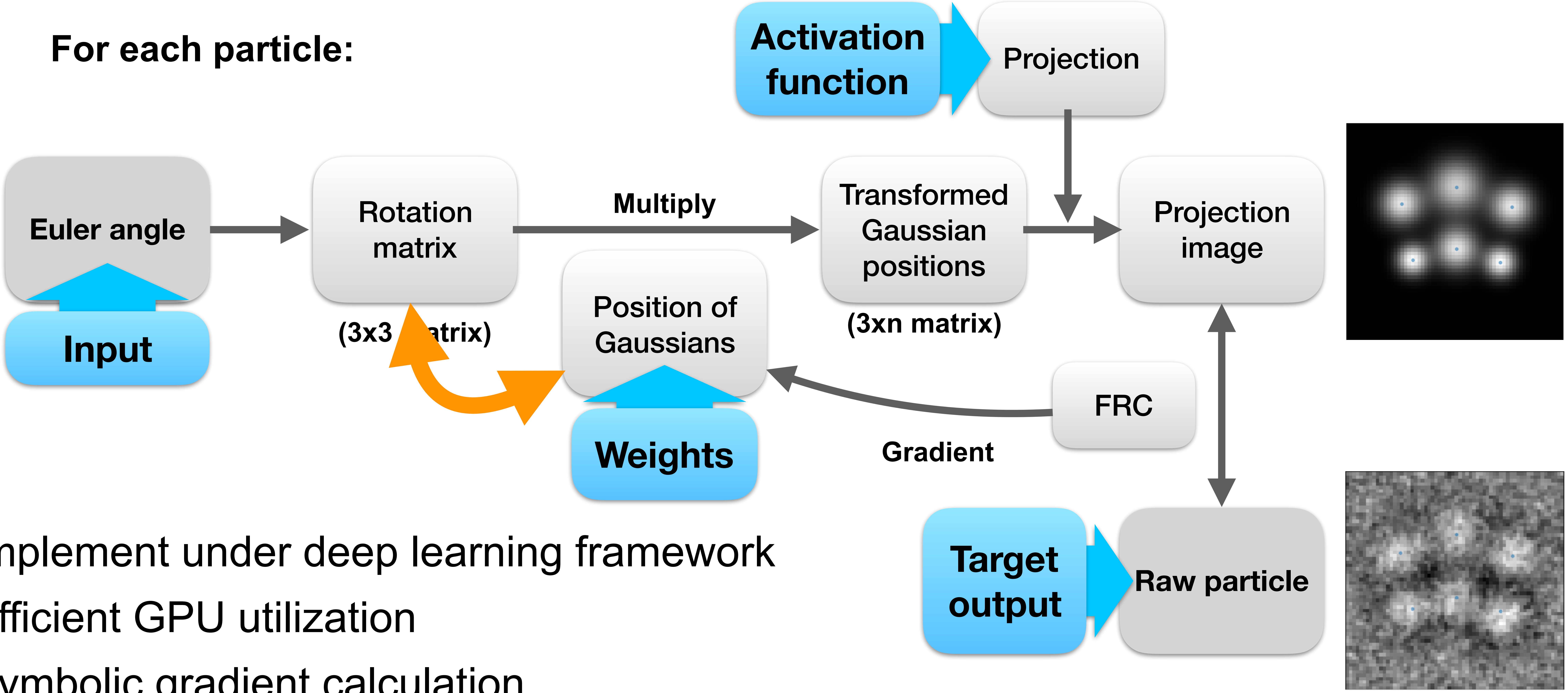
Linear/nonlinear motion



Both are linear in Gaussian representation.

Relationship with neural network

For each particle:



- Implement under deep learning framework
- Efficient GPU utilization
- Symbolic gradient calculation