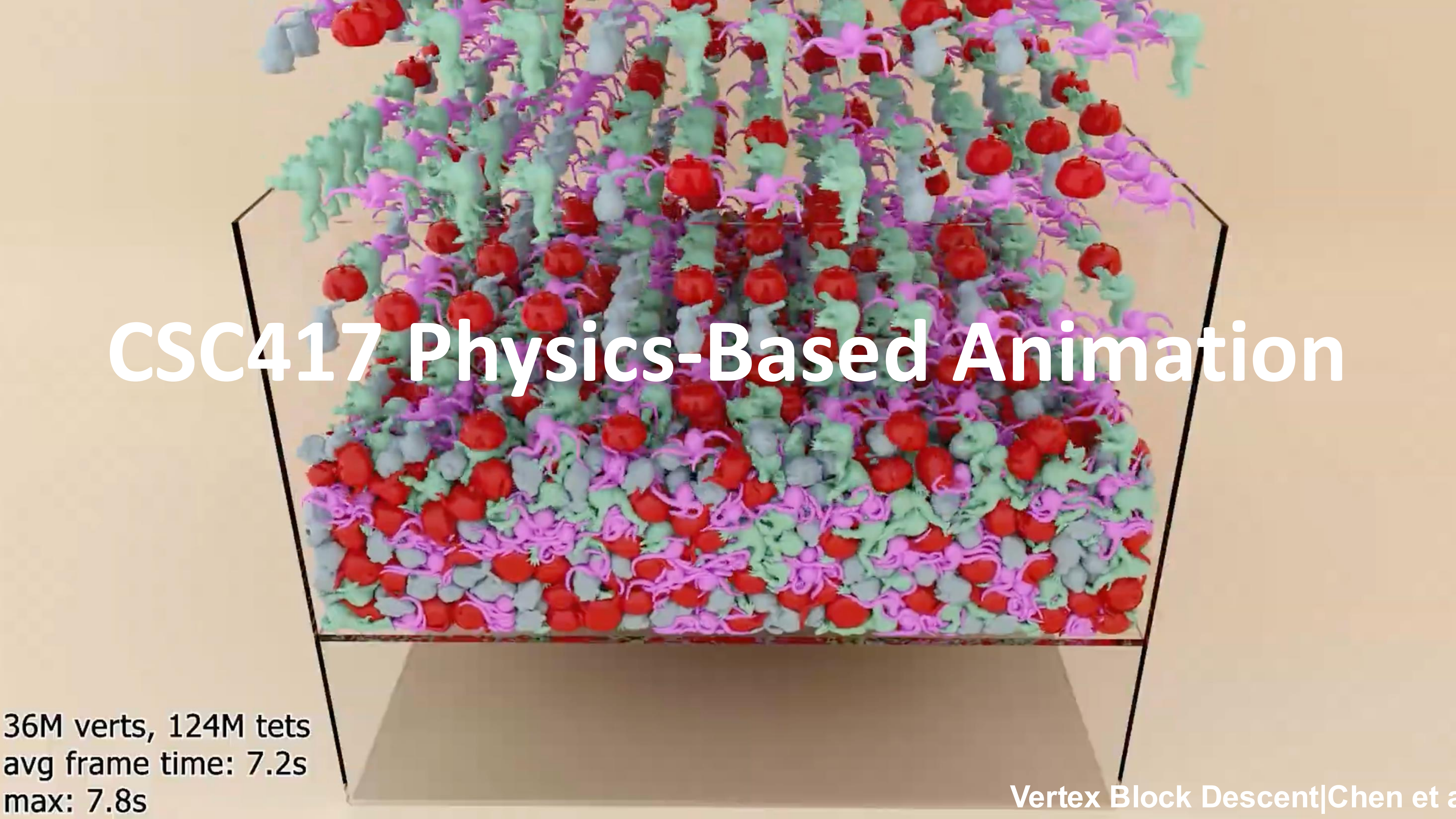
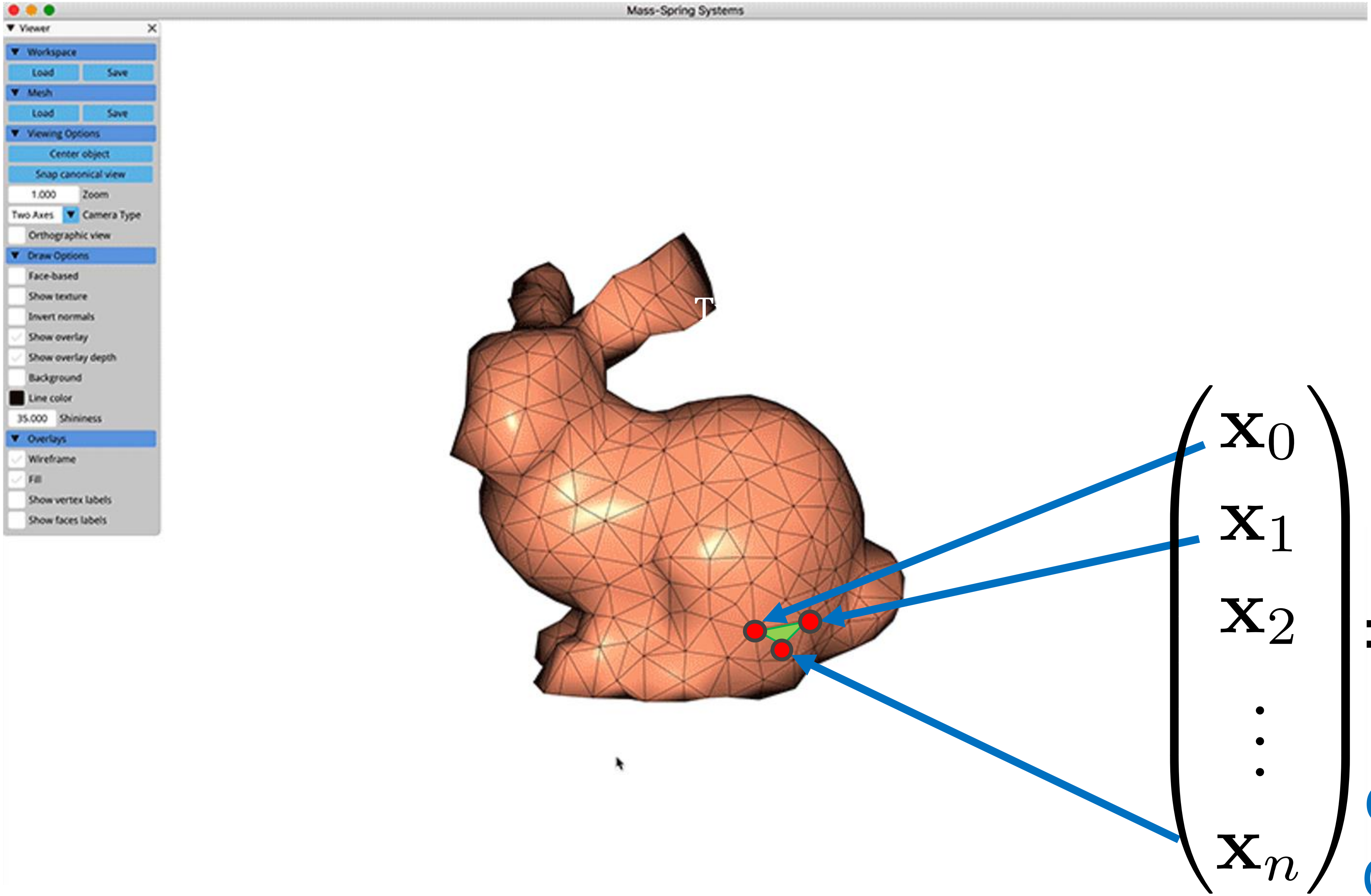
CSC417 Physics-Based Animation



Last Week: Reduced-Order Methods



$$\begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix} : q(0) + U\boldsymbol{r}$$

Generalized Coordinates

Questions from Previous Lecture ?

This Week: Fast Simulation via Coordinate Descent

We Solve This Every Time Step

$$E\left(\mathbf{q}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}^{\mathbf{i}}\right)^{\mathrm{T}} M\left(\mathbf{q}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}^{\mathbf{i}}\right) + h^2 V\left(\mathbf{q}^{\mathrm{i+1}}\right)$$

Gradient of what equals this ? Let's guess, then check

Newton's Method

Choose an initial guess

$$i = 0$$
$$\mathbf{v}^0 = \text{something}$$

Repeat until converged

Check for convergence

$$\left\| \frac{\partial E}{\partial \mathbf{v}}\Big|_{\mathbf{v}^i} \right\| < \text{tol}$$

Choose search direction

$$\mathrm{H}^i \mathbf{d} = -\mathbf{g}^i$$

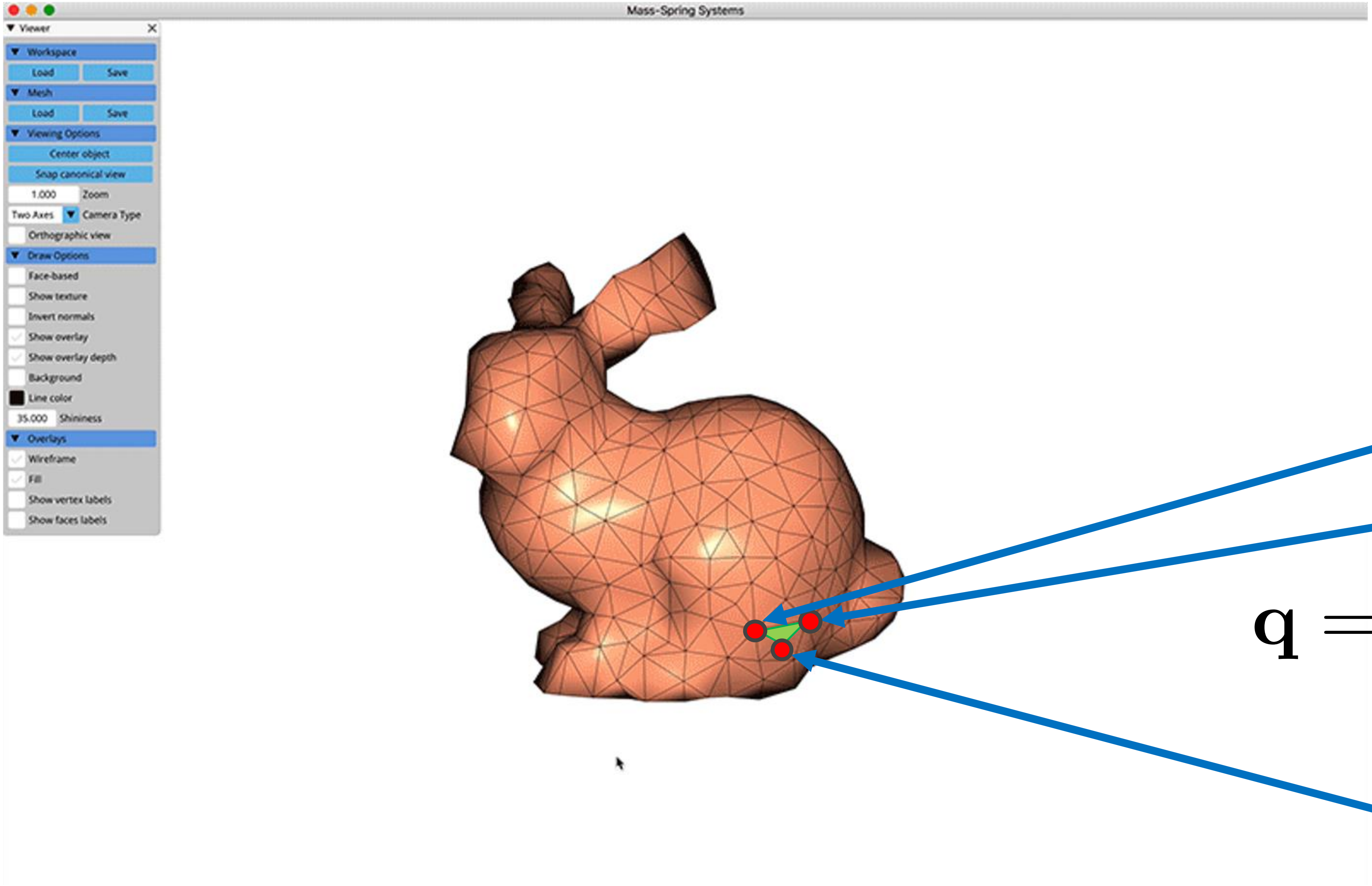
Solve linear system to get $\mathbf{d}$

Choose α using line search

Use search direction to update current guess

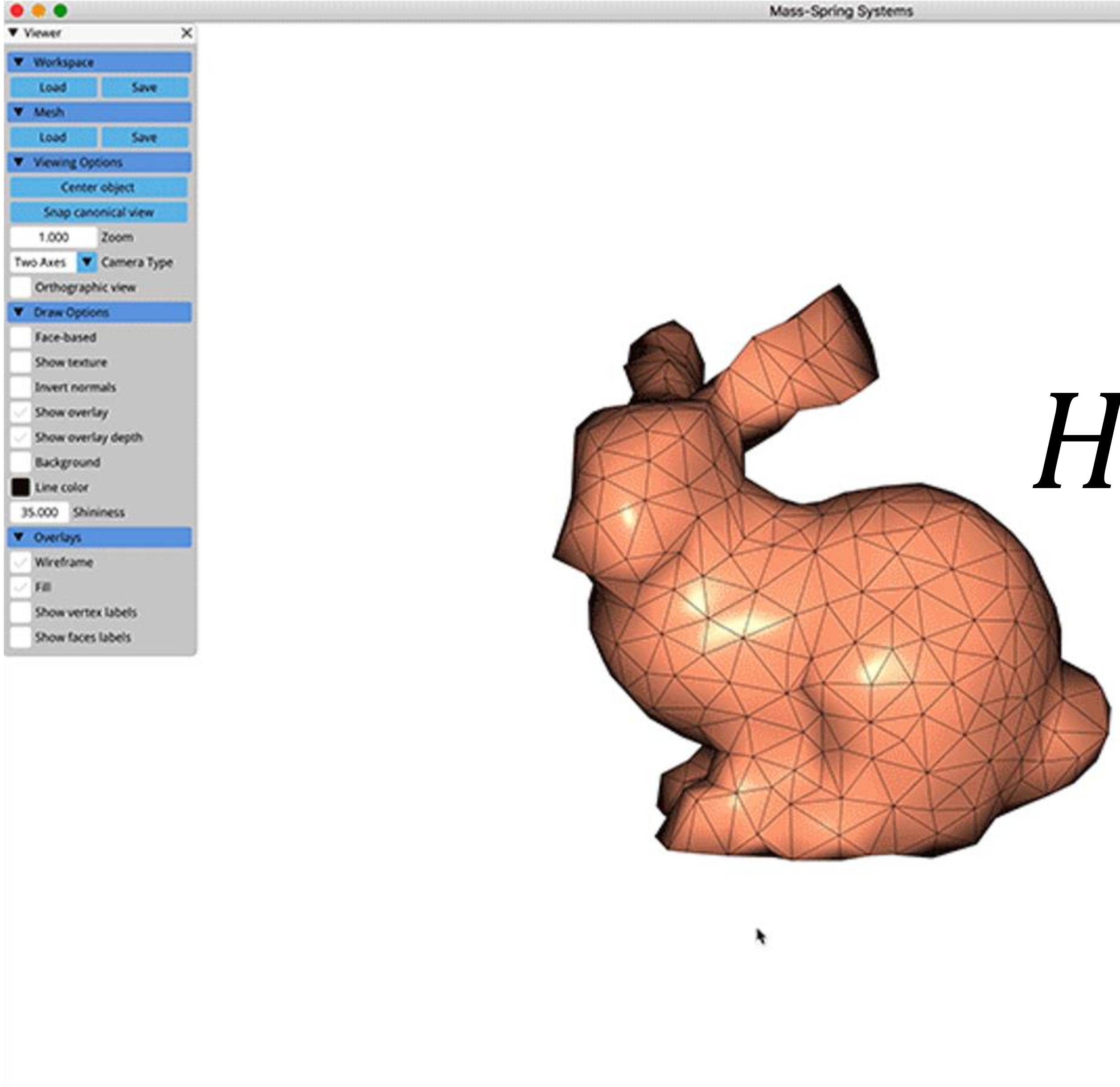
$$\mathbf{v}^{i+1} = \mathbf{v}^i + \alpha \mathbf{d}$$
$$i = i + 1$$

Spatial Discretization -- Finite Elements



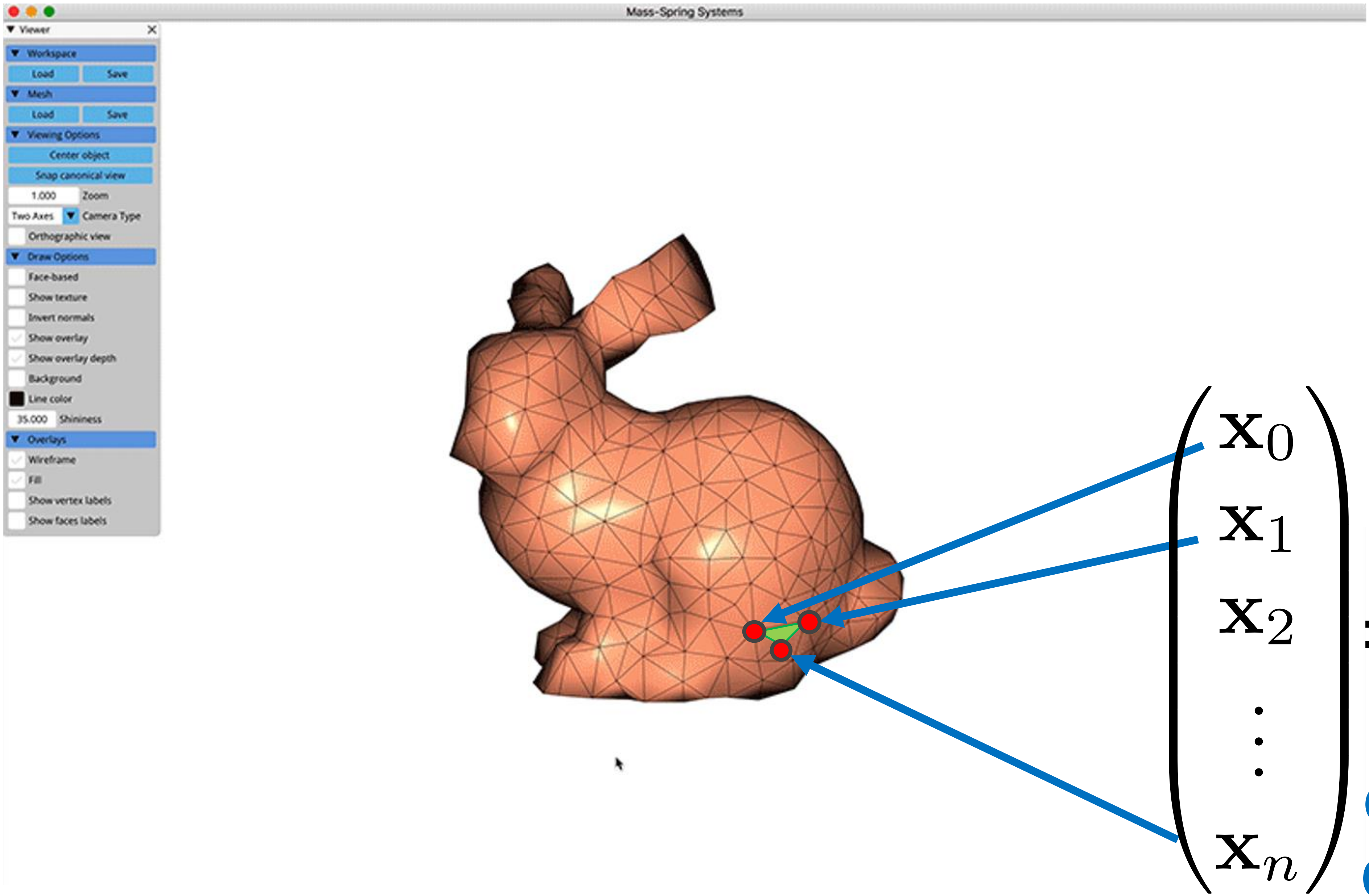
$$\mathbf{q} = \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix}$$

Assembly still visits every element ☹



$$H = \sum_t \underbrace{E_i^T H_i E_i}_{\text{Per-tetrahedron Hessian}}$$

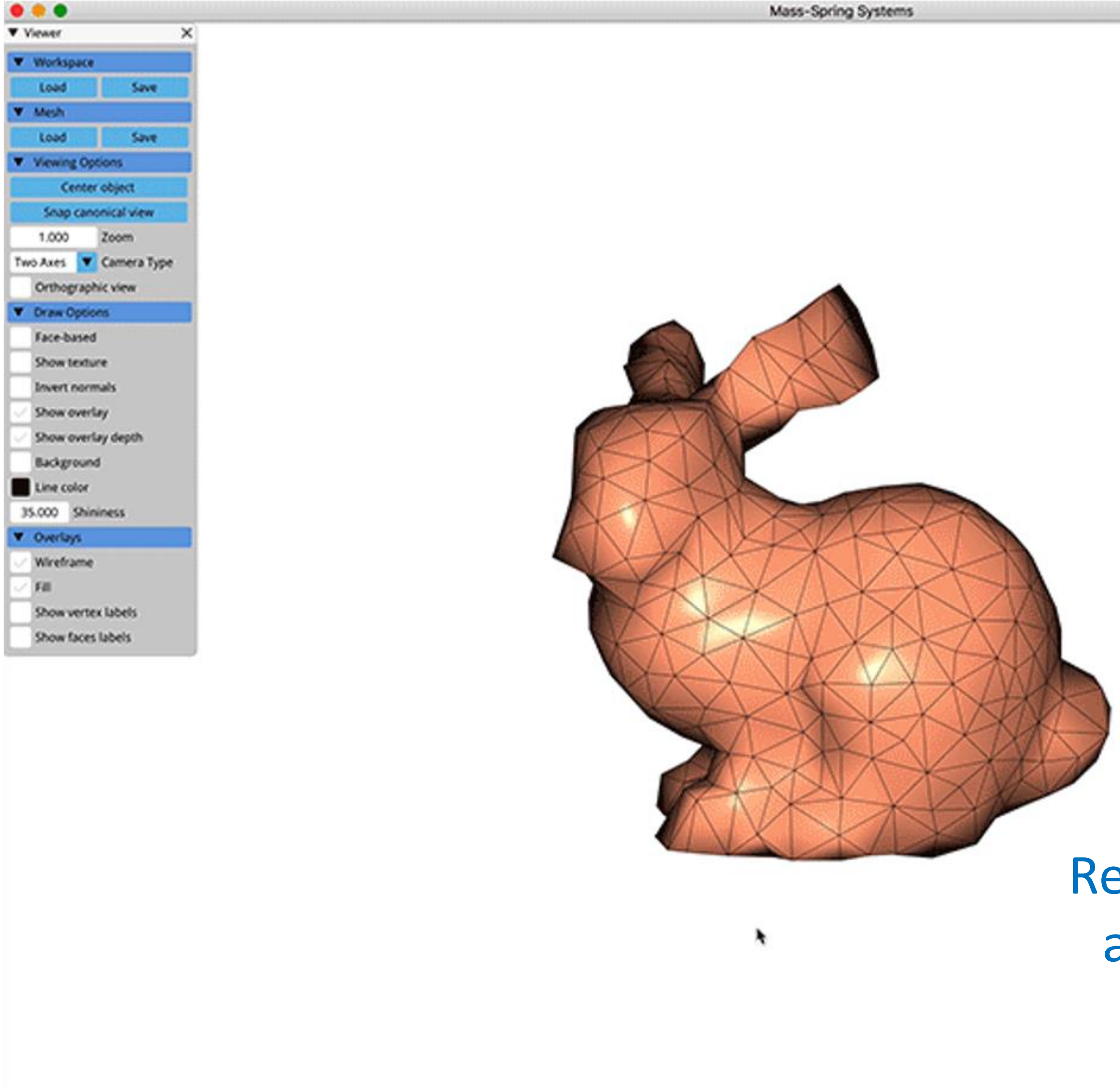
Last Week: Reduced-Order Methods



$$\begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix} : q(0) + U\boldsymbol{r}$$

Generalized Coordinates

Optimal Quadrature



$$H = \sum_{\mathrm{i} \in \mathrm{C}} \mathrm{w_i} U^T H_i U$$

Replace this big sum with a small sum over a few tetrahedra in the cubuature set C.

Another way to go fast

1. Avoid reducing the simulation space
2. Split up problem in a way that allows us to exploit parallelism and avoid building large matrices

Re-thinking Energy Minimization

$$E\left(\mathbf{q}^{\mathbf{i+1}}\right) = \underbrace{\frac{1}{2}\left(\mathbf{q}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}^{\mathbf{i}}\right)^{\mathrm{T}} M\left(\mathbf{q}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}^{\mathbf{i}}\right)}_{\text{Global “Inertial” Energy}} + \overbrace{h^2 V\left(\mathbf{q}^{\mathbf{i+1}}\right)}^{\text{Global Potential Energy}}$$

Newton's Method

Repeat until converged

Choose an initial guess

$$i = 0$$
$$\mathbf{v}^0 = \text{something}$$

Check for convergence

$$\left|\left|\frac{\partial E}{\partial \mathbf{v}}\right|_{\mathbf{v}^i}\right|\right| < \text{tol}$$

Choose search direction

$$\mathrm{H}^i\mathbf{d} = -\mathbf{g}^i$$

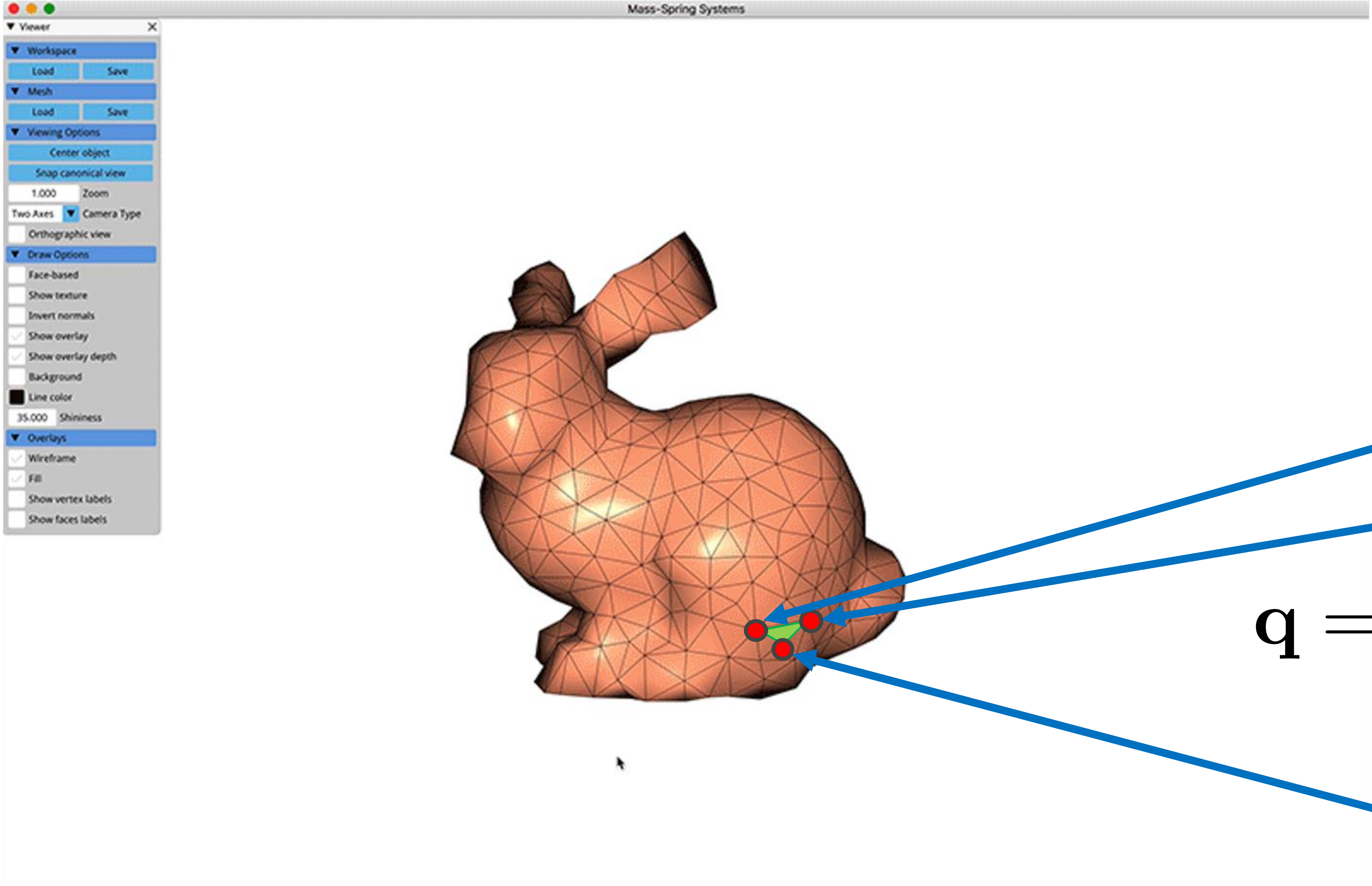
Large Global Sparse System

Choose α using line search

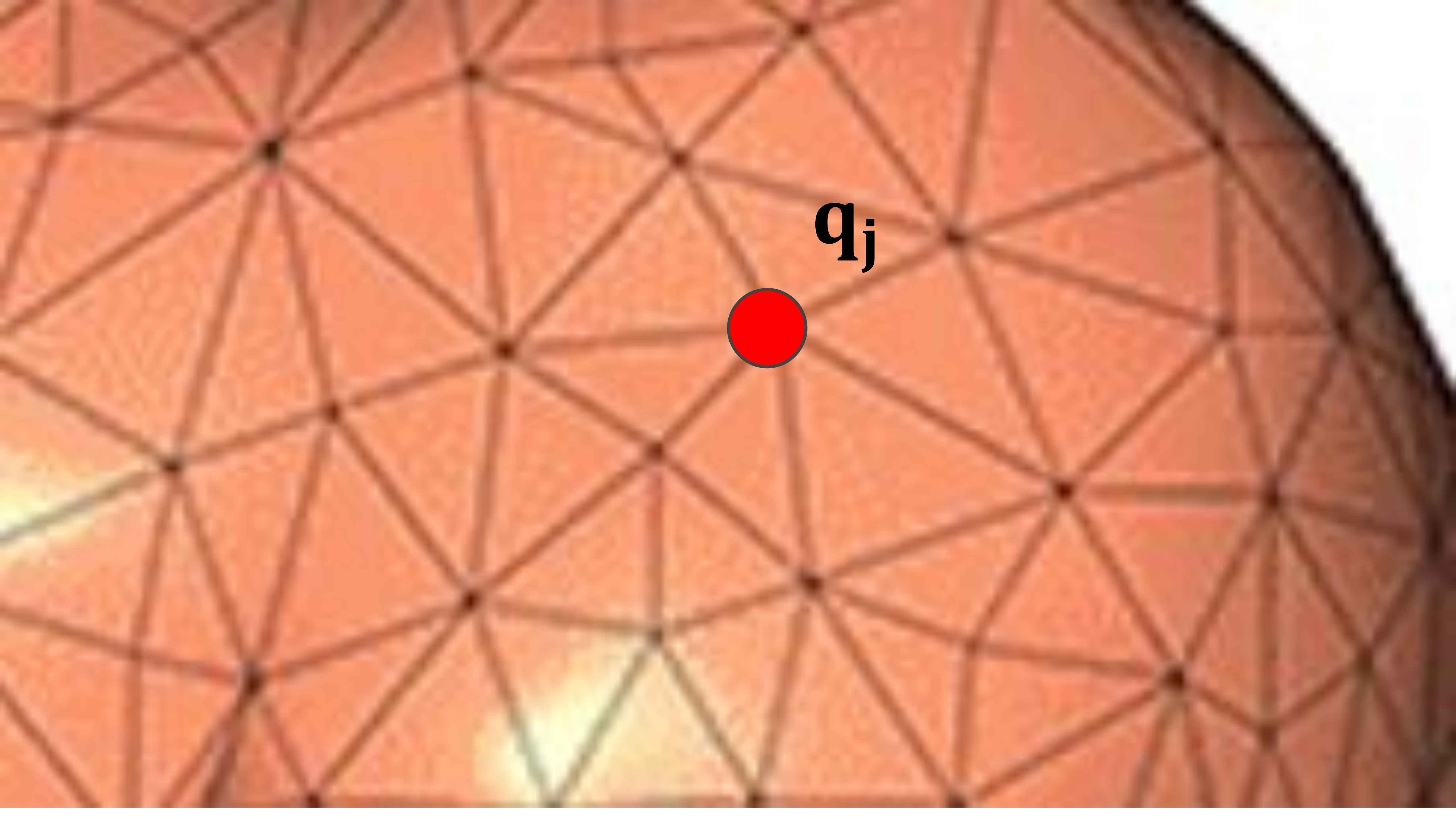
Use search direction to update current guess

$$\mathbf{v}^{i+1} = \mathbf{v}^i + \alpha\mathbf{d}$$
$$i = i + 1$$

Coordinate Descent

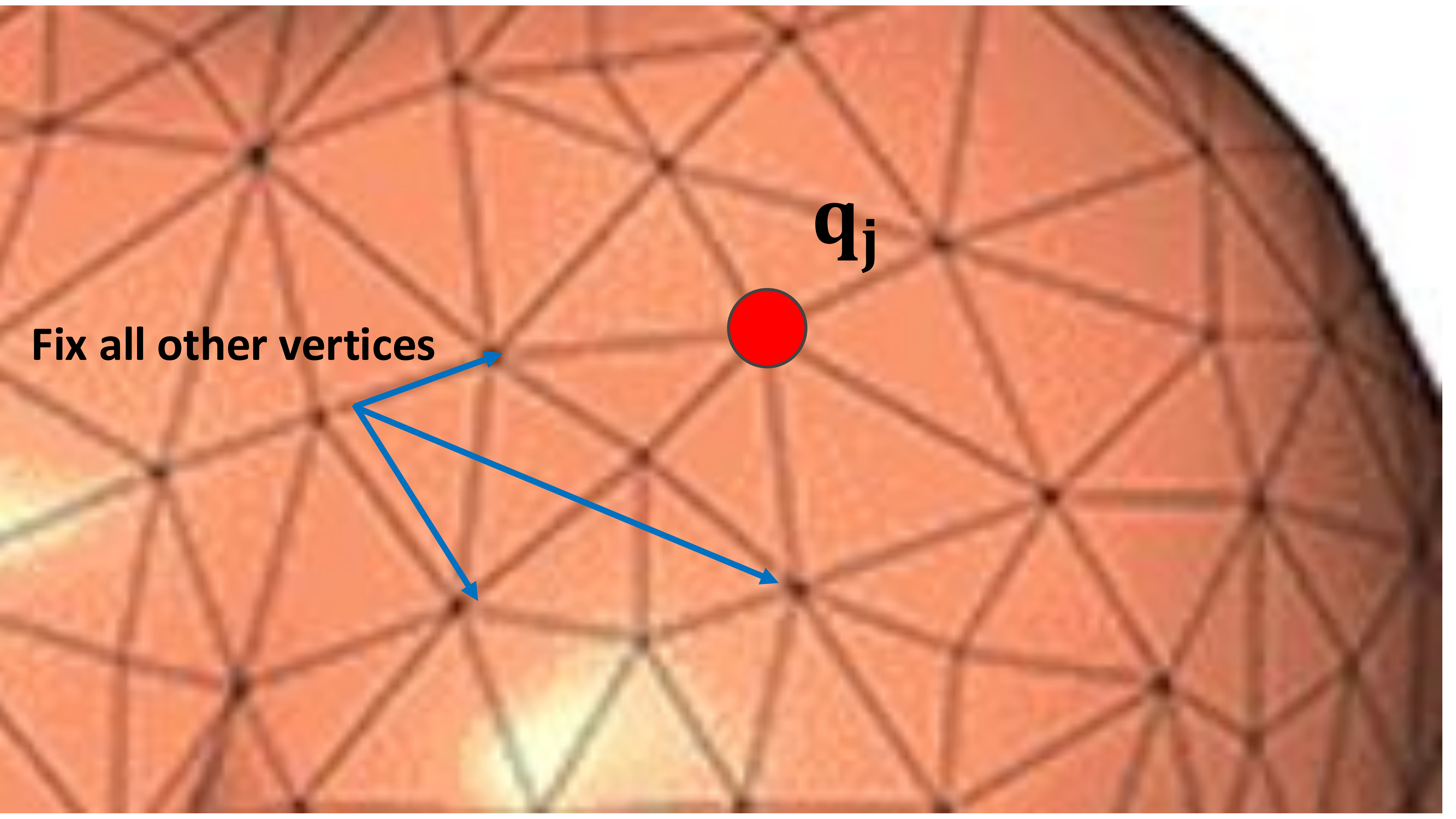


$$\mathbf{q} = \begin{pmatrix} \mathbf{x}_0 \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_n \end{pmatrix}$$


q_j

What happens if we just minimize wrt $\mathbf{q}_\mathbf{j}$?

$$E\left(\mathbf{q}^{\mathbf{i+1}}\right)=\underbrace{\frac{1}{2}\left(\mathbf{q}^{\mathbf{i+1}}-\tilde{\mathbf{q}}^{\mathbf{i}}\right)^{\mathrm{T}}M\left(\mathbf{q}^{\mathbf{i+1}}-\tilde{\mathbf{q}}^{\mathbf{i}}\right)}_{\text{Global “Inertial” Energy}}+\overbrace{h^{2}V\left(\mathbf{q}^{\mathbf{i+1}}\right)}^{\text{Global Potential Energy}}$$



$\mathbf{q_j}$
Fix all other vertices

What happens if we just minimize wrt $\mathbf{q}_j$?

$$E(\mathbf{q}^{i+1}) = \underbrace{\frac{1}{2}(\mathbf{q}^{i+1} - \tilde{\mathbf{q}}^{i})^{T} M(\mathbf{q}^{i+1} - \tilde{\mathbf{q}}^{i})}_{\text{Global “Inertial” Energy}} + \overbrace{h^2 V(\mathbf{q}^{i+1})}^{\text{Global Potential Energy}}$$

Inertial Energy Structure

$$\frac{1}{2}\left(\mathbf{q}_0{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_0^{\mathrm{i}}, \mathbf{q}_1{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_1^{\mathrm{i}}, \dots\right)
\begin{array}{c c}
 & \begin{array}{ccc} \text{Vertex 0} & \text{Vertex 1} & \text{Vertex 2} \end{array} \\
\begin{array}{c} \text{Vertex 0} \\ \text{Vertex 1} \\ \text{Vertex 2} \end{array} &
\begin{bmatrix} M_{00} & M_{01} & M_{02} \\ \dots & M_{11} & M_{12} \\ \dots & \dots & M_{22} \end{bmatrix}
\end{array}
\begin{pmatrix} \mathbf{q}_0{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_0^{\mathrm{i}} \\ \mathbf{q}_1{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_1^{\mathrm{i}} \\ \dots \end{pmatrix}$$

Inertial Energy Structure

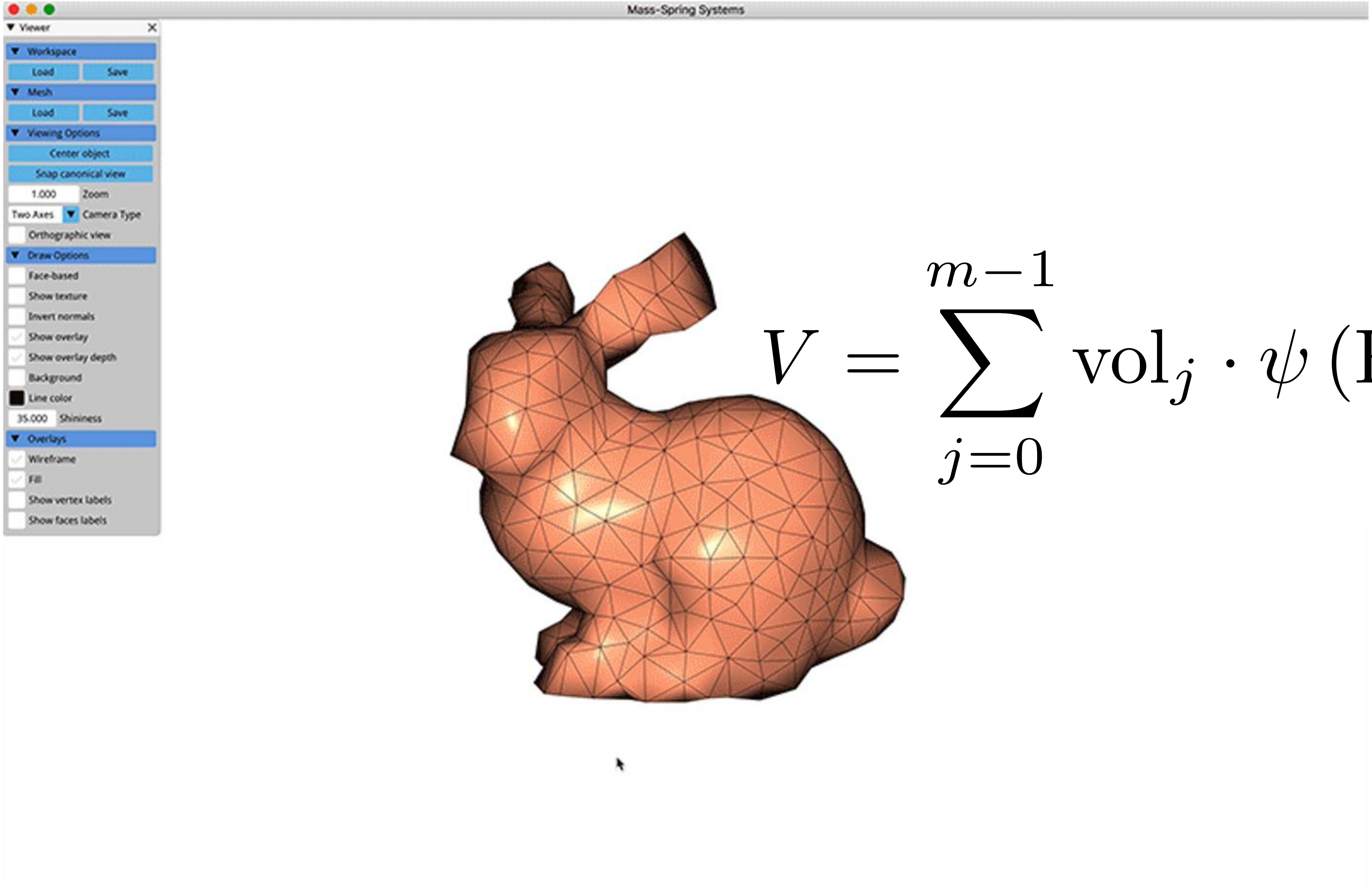
$$\frac{1}{2}\left(\mathbf{q}_0{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_0^{\mathrm{i}}\right)^T M_{00}\left(\mathbf{q}_0{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_0^{\mathrm{i}}\right)$$

$$+\left(\mathbf{q}_1{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_1^{\mathrm{i}}\right)^T M_{01}\left(\mathbf{q}_0{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_0^{\mathrm{i}}\right)$$

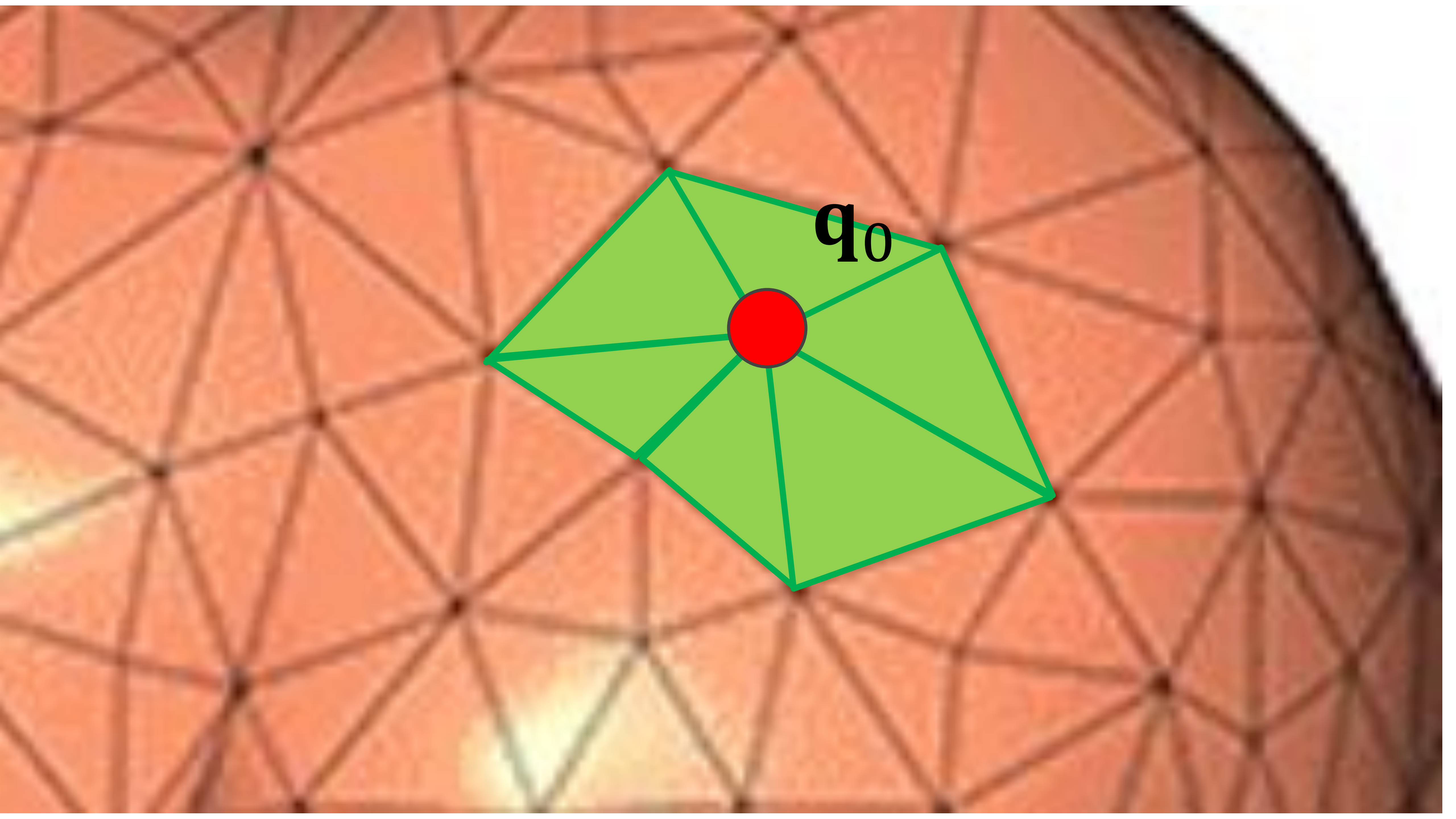
$$+\left(\mathbf{q}_2{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_2^{\mathrm{i}}\right)^T M_{02}\left(\mathbf{q}_0{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_0^{\mathrm{i}}\right)$$

$+$ Bunch of terms that don’t depend on $\mathbf{q}_0$

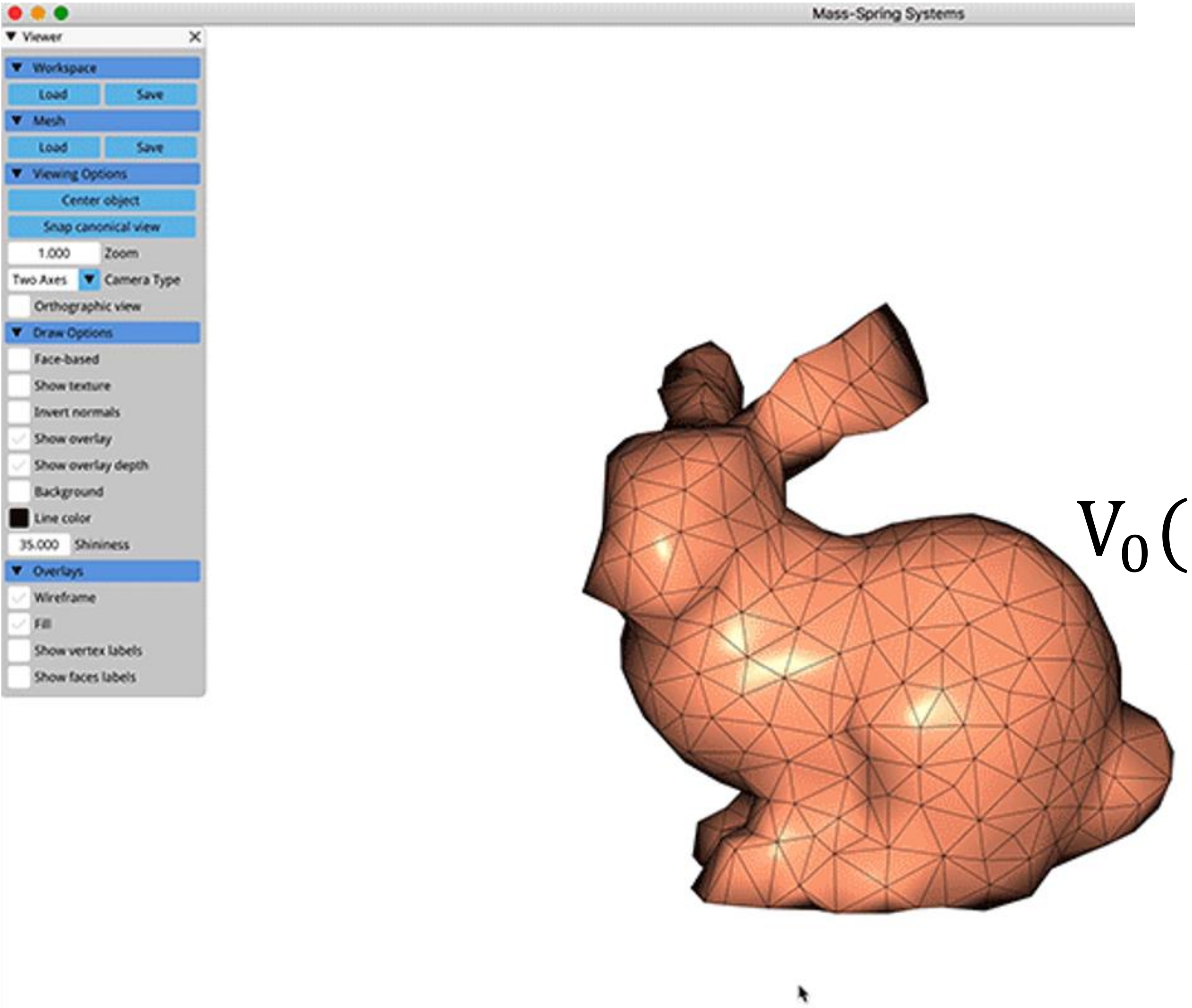
Potential Energy Structure (Way back in Lectures 2 and 3)



$$V = \sum_{j=0}^{m-1} \mathrm{vol}_j \cdot \psi\left(\mathrm{F}_j\left(\mathrm{E}_j \mathbf{q}\right)\right)$$


$\mathbf{q}_0$

Potential Energy Structure (Way back in Lectures 2 and 3)



$$\mathrm{V}_0(\mathbf{q}_0{}^{\mathbf{i+1}}) = \sum_{\mathrm{i}\,\in\mathcal{N}} vol_i\, \psi(F_i(\mathbf{q}_0{}^{\mathbf{i+1}}, \dots))$$

Tetrahedra containing $\mathbf{q}_0$

Other vertices of the tetrahedron

For a single vertex

$$E(\mathbf{q}_j{}^{\mathbf{i+1}}) = \frac{1}{2}(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}})^T M_{\mathrm{jj}}(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}) + \mathbf{p}_j{}^T(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}) + h^{\mathbf{2}} V_j(\mathbf{q}_j{}^{\mathbf{i+1}})$$

Momentum from other degrees-of-freedom

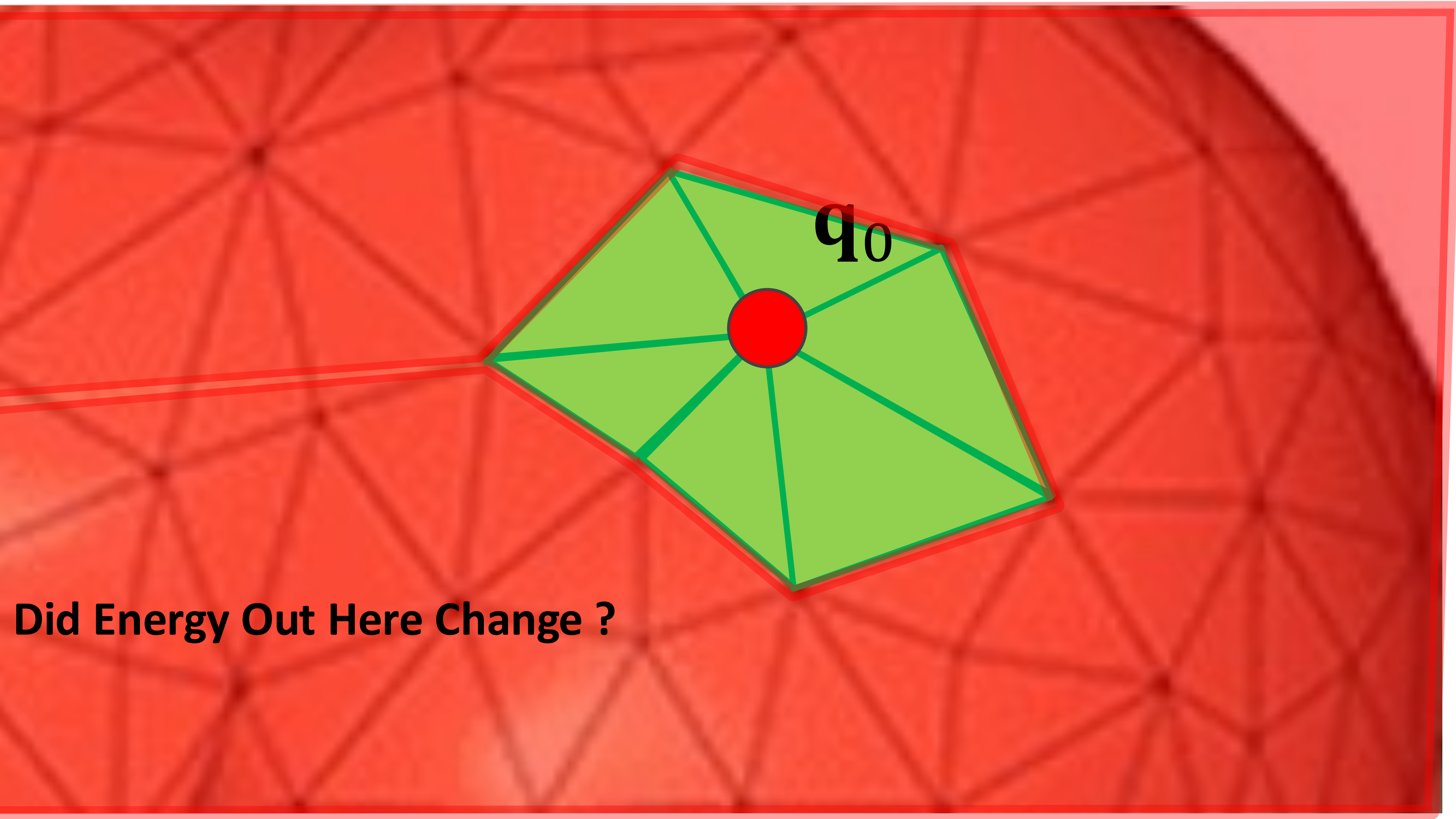
Inertial Energy Structure

$$\mathbf{p}_0 = \begin{array}{cc} & \begin{array}{cc} \text{Vertex 1} & \text{Vertex 2} \end{array} \\ \text{Vertex 0} & \begin{bmatrix} M_{01} & M_{02} \end{bmatrix} \end{array} \begin{pmatrix} \mathbf{q}_1{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_1^{\mathrm{i}} \\ \mathbf{q}_2{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_2^{\mathrm{i}} \\ \dots \end{pmatrix}$$

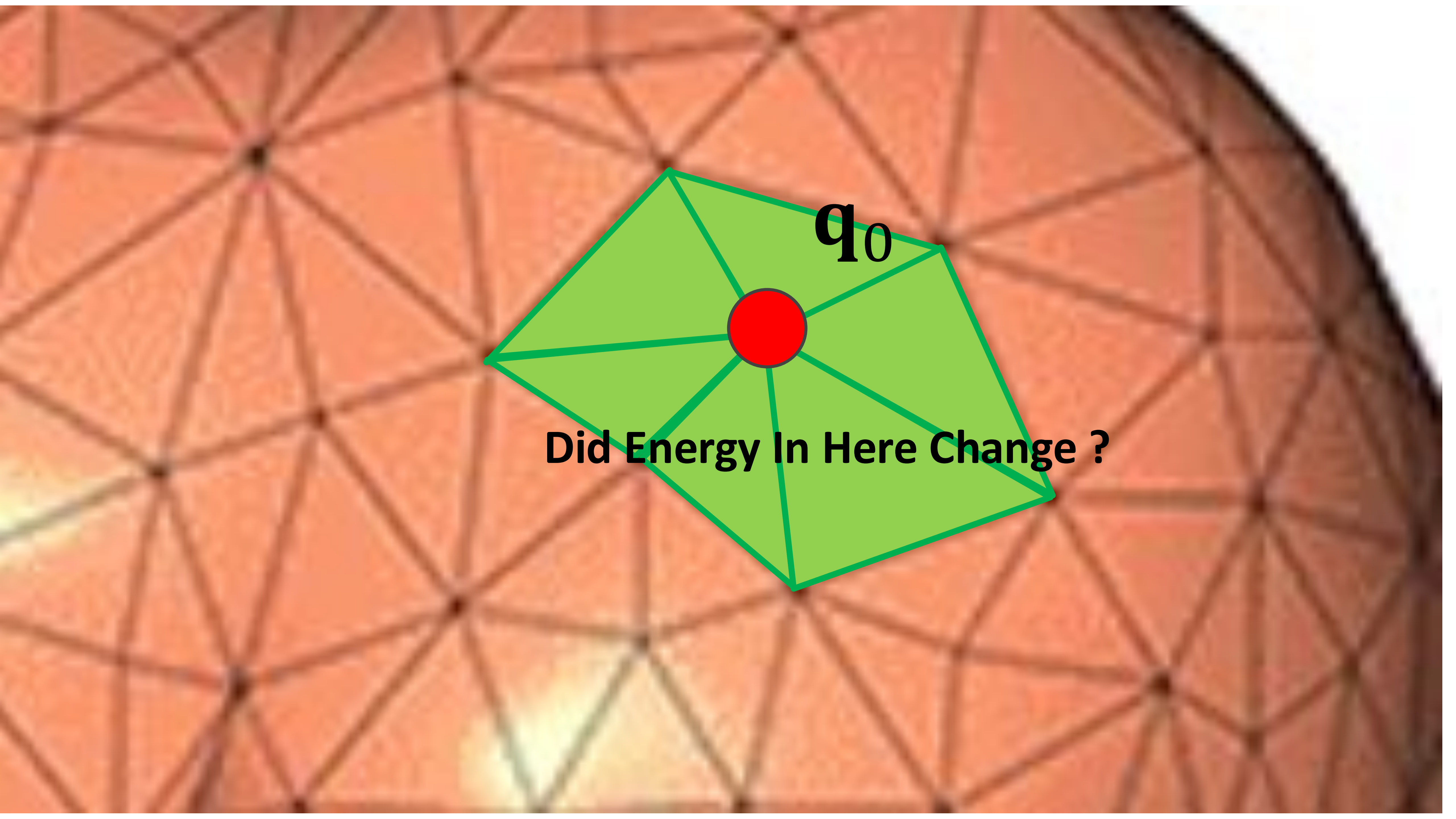
For a single vertex

$$E\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right)^T M_{\mathrm{jj}}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + \mathbf{p}_j{}^T\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + h^{\mathbf{2}} V_j\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right)$$

Momentum from other degrees-of-freedom

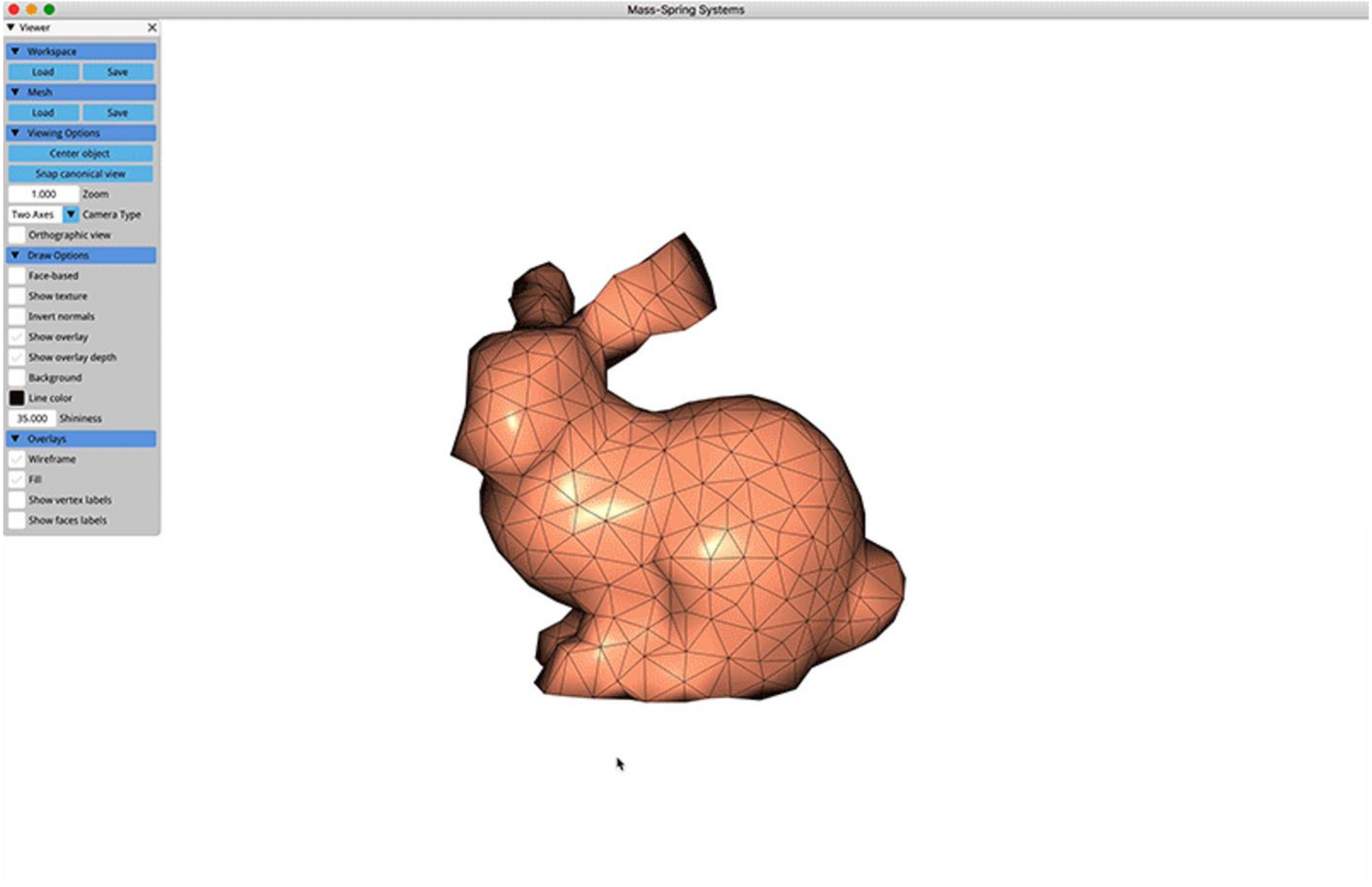


Did Energy Out Here Change ?



$\mathbf{q}_0$
Did Energy In Here Change ?

What happened to total energy then ?



So this suggests a new minimization strategy

Repeat for awhile

For each vertex, j

Minimize

$$E\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right)^T M_{\mathrm{jj}}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + \mathbf{p}_j{}^T\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + h^{\mathbf{2}} V_j\left(\mathbf{q}_j{}^{\mathrm{i+1}}\right)$$

Why is this fast ?

Repeat for awhile

For each vertex, j

Minimize

$$E\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right)^T M_{\mathrm{jj}}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + \mathbf{p}_j{}^T\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + h^{\mathbf{2}} V_j\left(\mathbf{q}_j{}^{\mathrm{i+1}}\right)$$

Why is this fast ?

Why is this fast ?

Repeat for awhile

For each vertex, j

Minimize

$$E(\mathbf{q}_j{}^{\mathbf{i+1}}) = \frac{1}{2}(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i})^T M_\mathrm{jj}(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}) + \mathbf{p}_j{}^T(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}) + h^2 V_j(\mathbf{q}_j{}^{\mathrm{i+1}})$$

Fast to solve using Newton’s Method

Newton's Method

Choose an initial guess

$$i = 0$$
$$\mathbf{v}^0 = \text{something}$$

Repeat until converged

Check for convergence

$$\left\| \left. \frac{\partial E}{\partial \mathbf{v}} \right|_{\mathbf{v}^i} \right\| < \text{tol}$$

Choose search direction

$$\mathrm{H}^i \mathbf{d} = -\mathbf{g}^i$$

Here's our main cost

Choose α using line search

Use search direction to update current guess

$$\mathbf{v}^{i+1} = \mathbf{v}^i + \alpha \mathbf{d}$$
$$i = i + 1$$

So this suggests a new minimization strategy

Repeat for awhile

For each vertex, j

Minimize

$$E(\mathbf{q}_j{}^{\mathbf{i+1}}) = \frac{1}{2}(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i})^T M_\mathrm{jj}(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}) + \mathbf{p}_j{}^T(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}) + h^{\mathbf{2}} V_j(\mathbf{q}_j{}^{\mathrm{i+1}})$$

$$\mathbf{q}_j{}^{\mathbf{i+1}} \in \mathcal{R}^3$$

$$\mathbf{g} = \frac{\partial E_j}{\partial \mathbf{q}_j{}^{\mathrm{i+1}}} \in \mathcal{R}^3$$

$$\mathrm{H} = \frac{\partial^2 E_j}{\partial \mathbf{q}_j{}^{\mathrm{i+1}} \partial \mathbf{q}_j{}^{\mathrm{i+1}}} \in \mathcal{R}^{3\times 3}$$

Newton's Method

Choose an initial guess

$$i = 0$$
$$\mathbf{v}^0 = \text{something}$$

Repeat until converged

Check for convergence

$$\left|\left|\frac{\partial E}{\partial \mathbf{v}}\Big|_{\mathbf{v}^i}\right|\right| < \text{tol}$$

Choose search direction

$$\mathrm{H}^i\mathbf{d} = -\mathbf{g}^i$$

Small, dense 3x3 linear system

Choose α using line search

Use search direction to update current guess

$$\mathbf{v}^{i+1} = \mathbf{v}^i + \alpha\mathbf{d}$$
$$i = i + 1$$

Why is this fast ?

Repeat for awhile

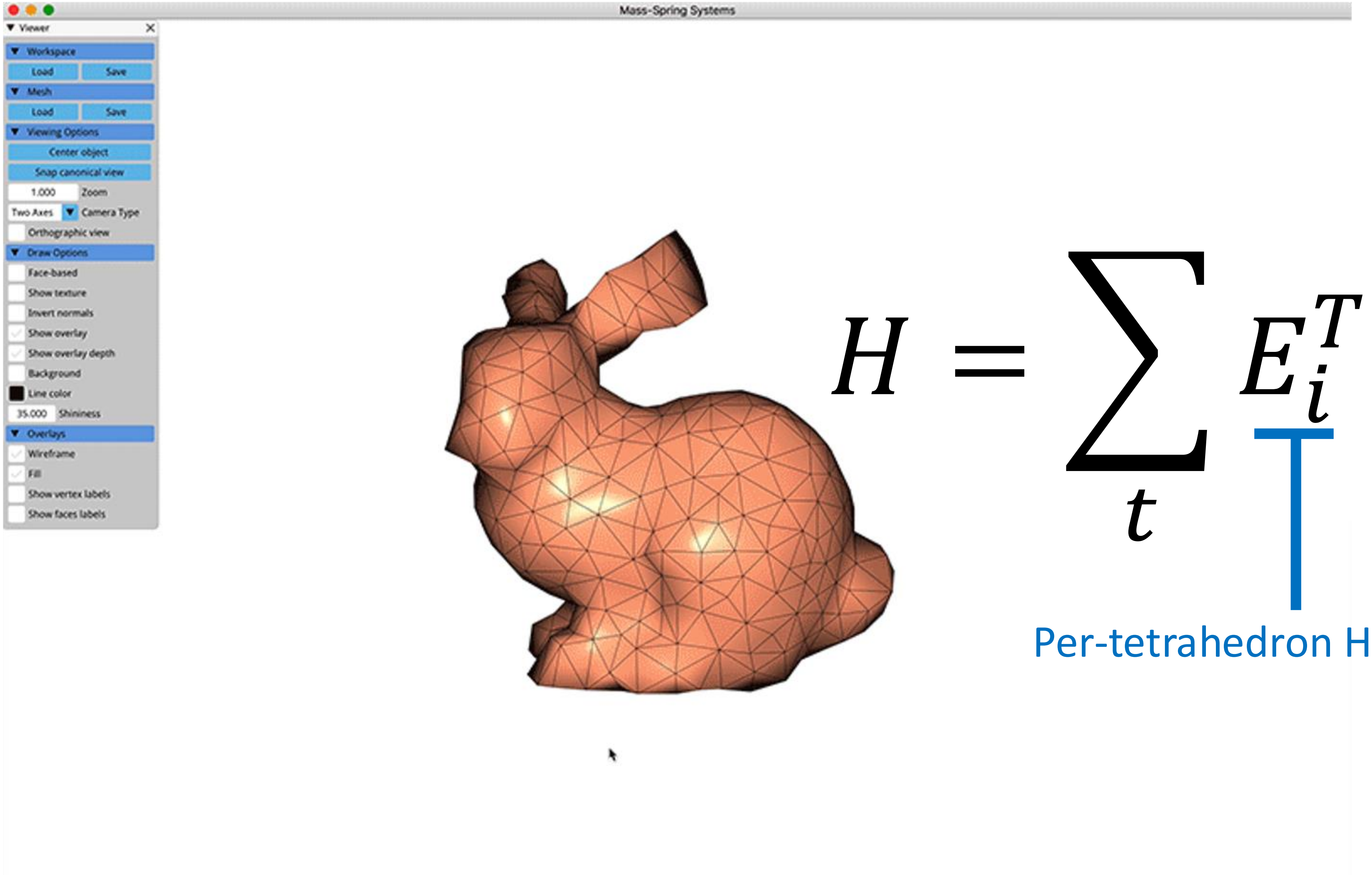
For each vertex, j

Minimize

$$E\left(\mathbf{q}_j^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_{\mathrm{j}}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right)^T M_{\mathrm{jj}}\left(\mathbf{q}_{\mathrm{j}}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + \mathbf{p}_j^{\,T}\left(\mathbf{q}_{\mathrm{j}}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + h^{\mathbf{2}} V_j\left(\mathbf{q}_j^{\mathrm{i+1}}\right)$$

Why is this fast ?

Assembly still visits every element ☹



$$H = \sum_t \underbrace{E_i^T H_i E_i}_{\text{Per-tetrahedron Hessian}}$$

Why is this fast ?

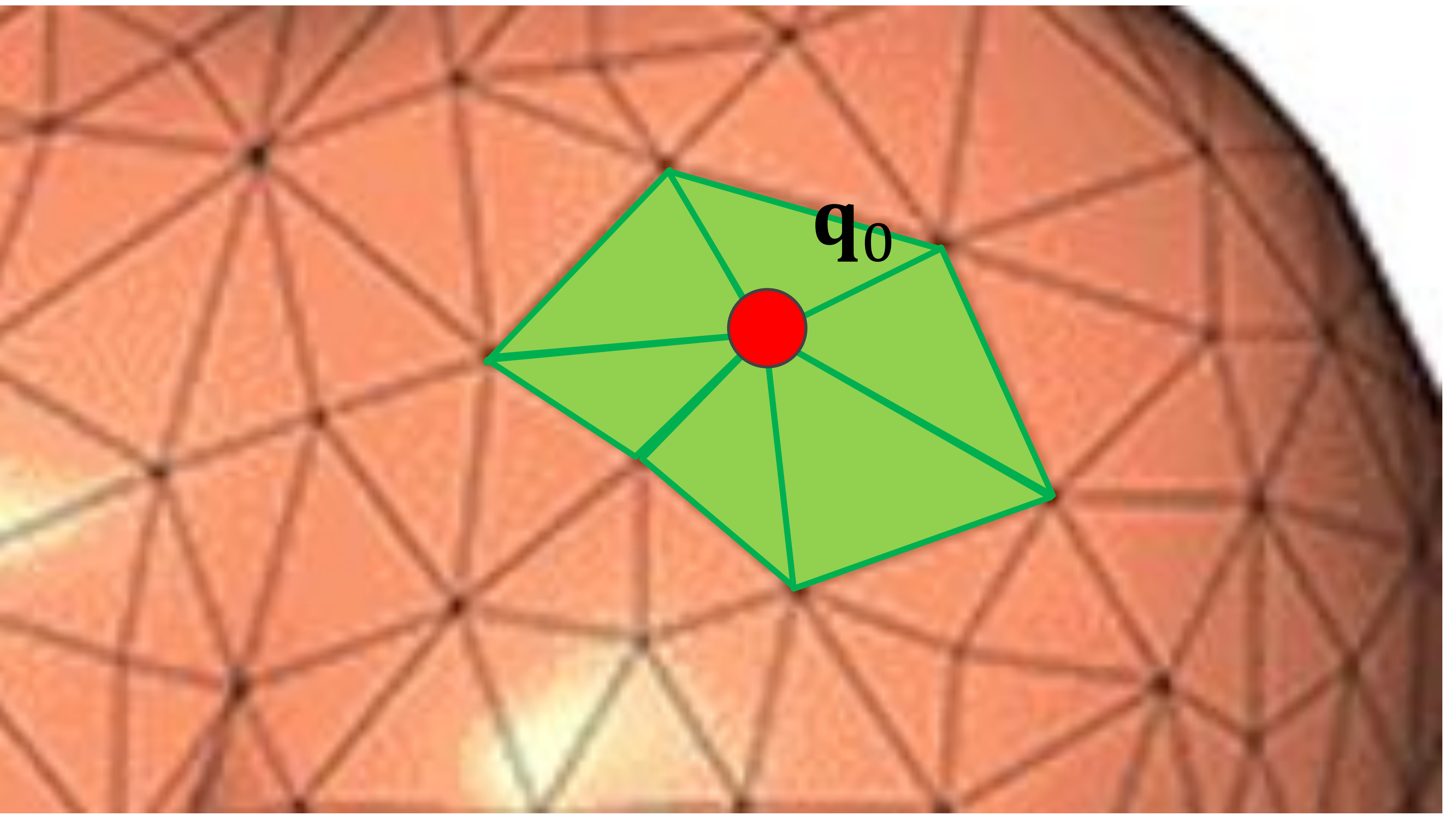
Repeat for awhile

For each vertex, j

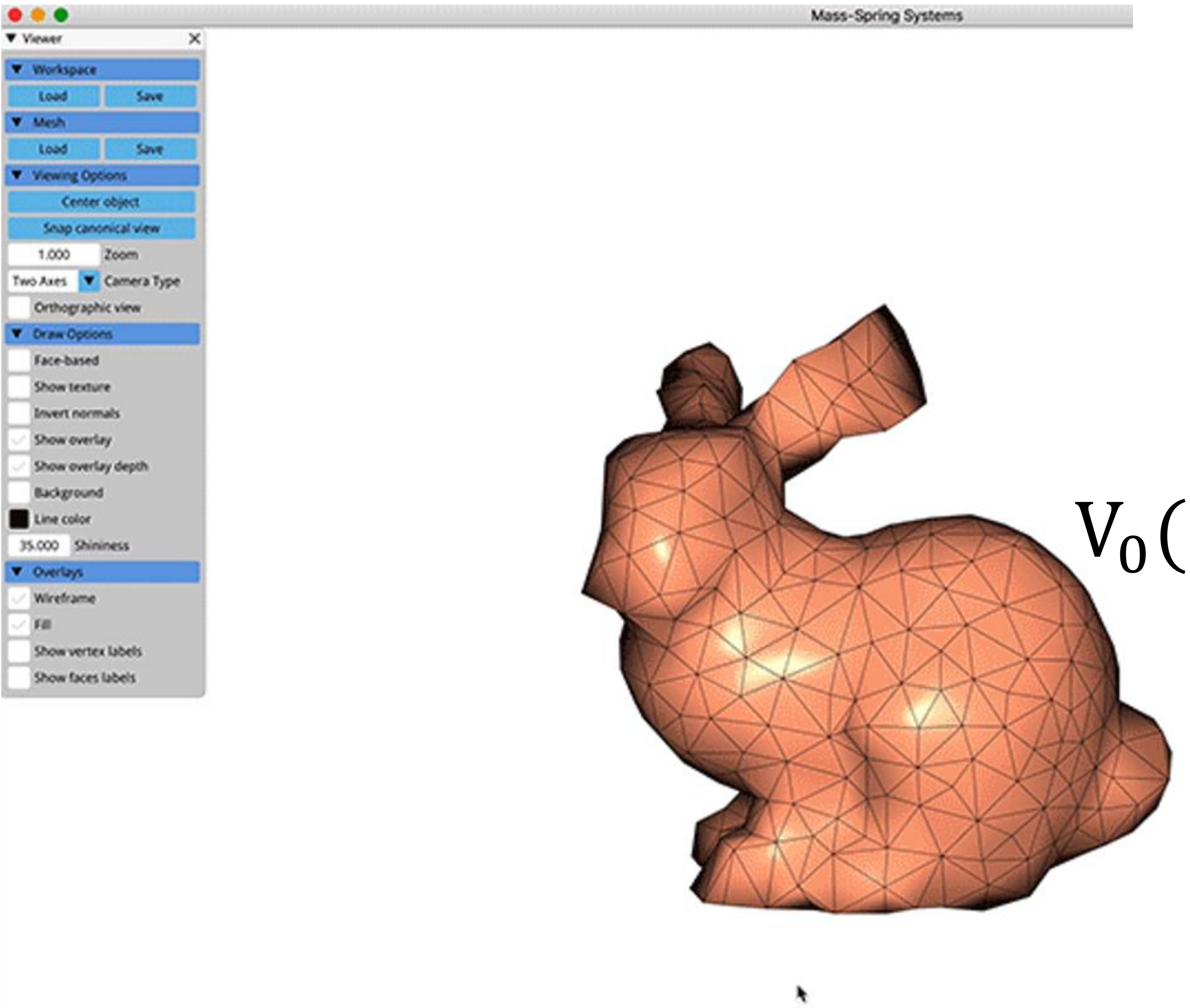
Minimize

$$E\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right)^T M_{\mathrm{jj}}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + \mathbf{p}_j{}^T\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + h^2 V_j\left(\mathbf{q}_j{}^{\mathrm{i+1}}\right)$$

Only depends on neighborhood of *j*


$\mathbf{q}_0$

Potential Energy Structure (Way back in Lectures 2 and 3)



$$\mathrm{V}_0(\mathbf{q}_0{}^{\mathbf{i+1}}) = \sum_{\mathrm{i} \in \mathcal{N}} vol_i \, \psi(F_i(\mathbf{q}_0{}^{\mathbf{i+1}}, \ldots))$$

Tetrahedra containing $\mathbf{q}_0$

Other vertices of the tetrahedron

Why is this fast ?

Repeat for awhile

For each vertex, j

Minimize

$$E\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}\right)^T M_\mathrm{jj}\left(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}\right) + \mathbf{p}_j{}^T\left(\mathbf{q}_\mathrm{j}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_\mathrm{j}^\mathrm{i}\right) + h^{\mathbf{2}} V_j\left(\mathbf{q}_j{}^{\mathrm{i+1}}\right)$$

Why is this fast ?

Parallelization

Repeat for awhile

For each vertex, j

Minimize

$$E(\mathbf{q}_j{}^{\mathbf{i+1}}) = \frac{1}{2}(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}})^T M_{\mathrm{jj}}(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}) + \mathbf{p}_j{}^T(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \widetilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}) + h^{\mathbf{2}} V_j(\mathbf{q}_j{}^{\mathrm{i+1}})$$

It's not obvious but we can do this in parallel

Parallelization

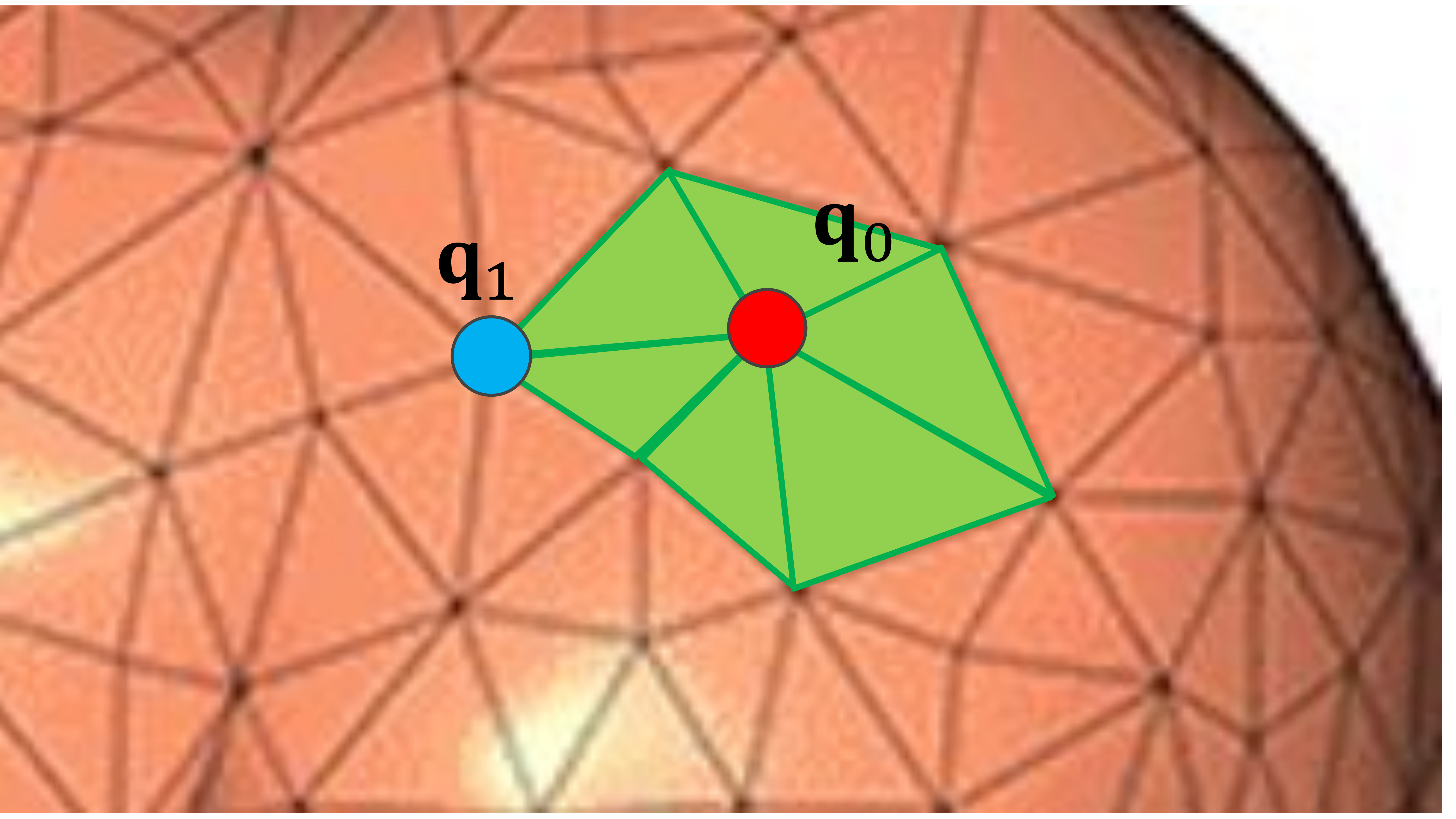
Repeat for awhile

For each vertex, j **Parallelize this Loop !!!**

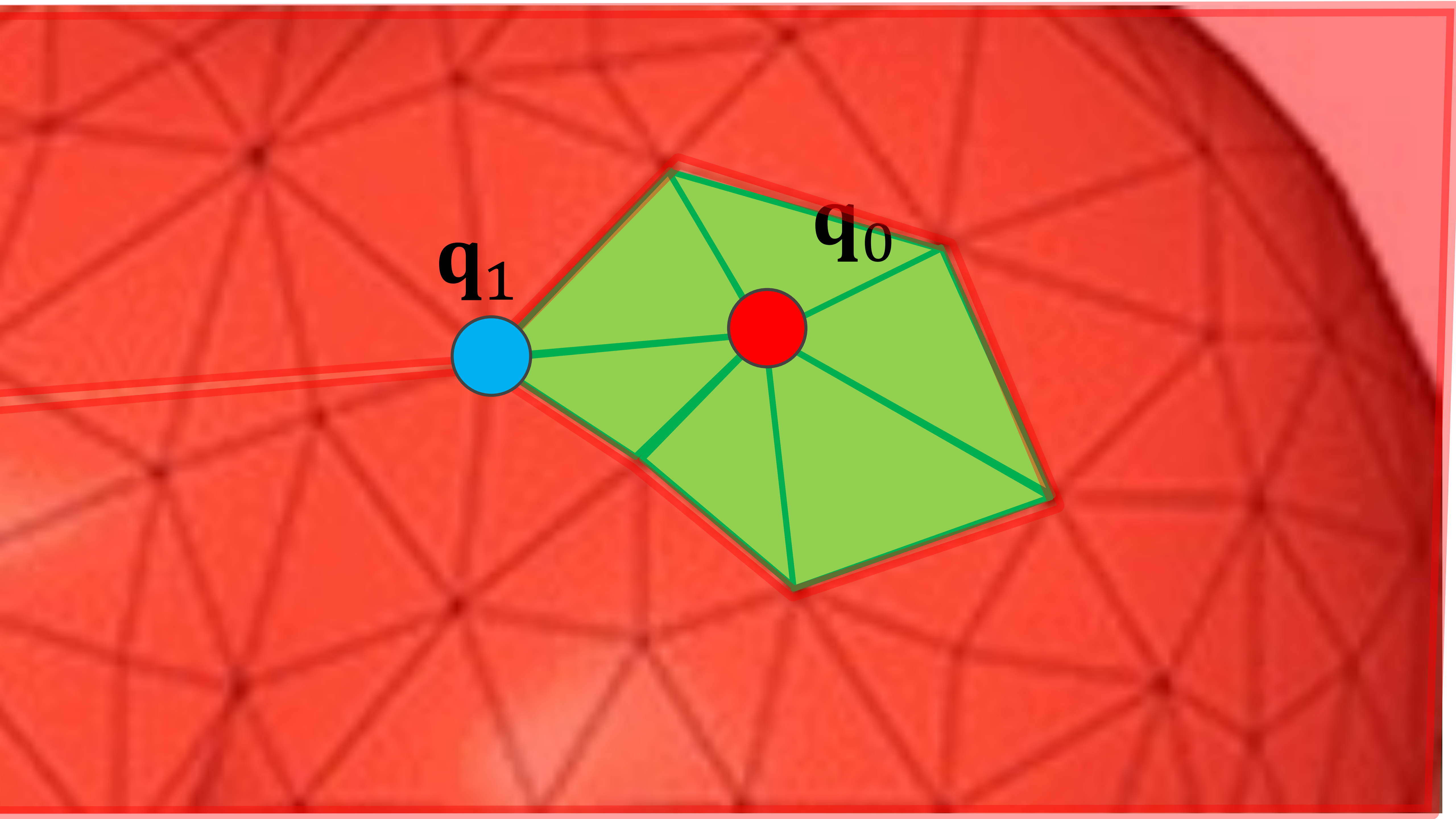
Minimize

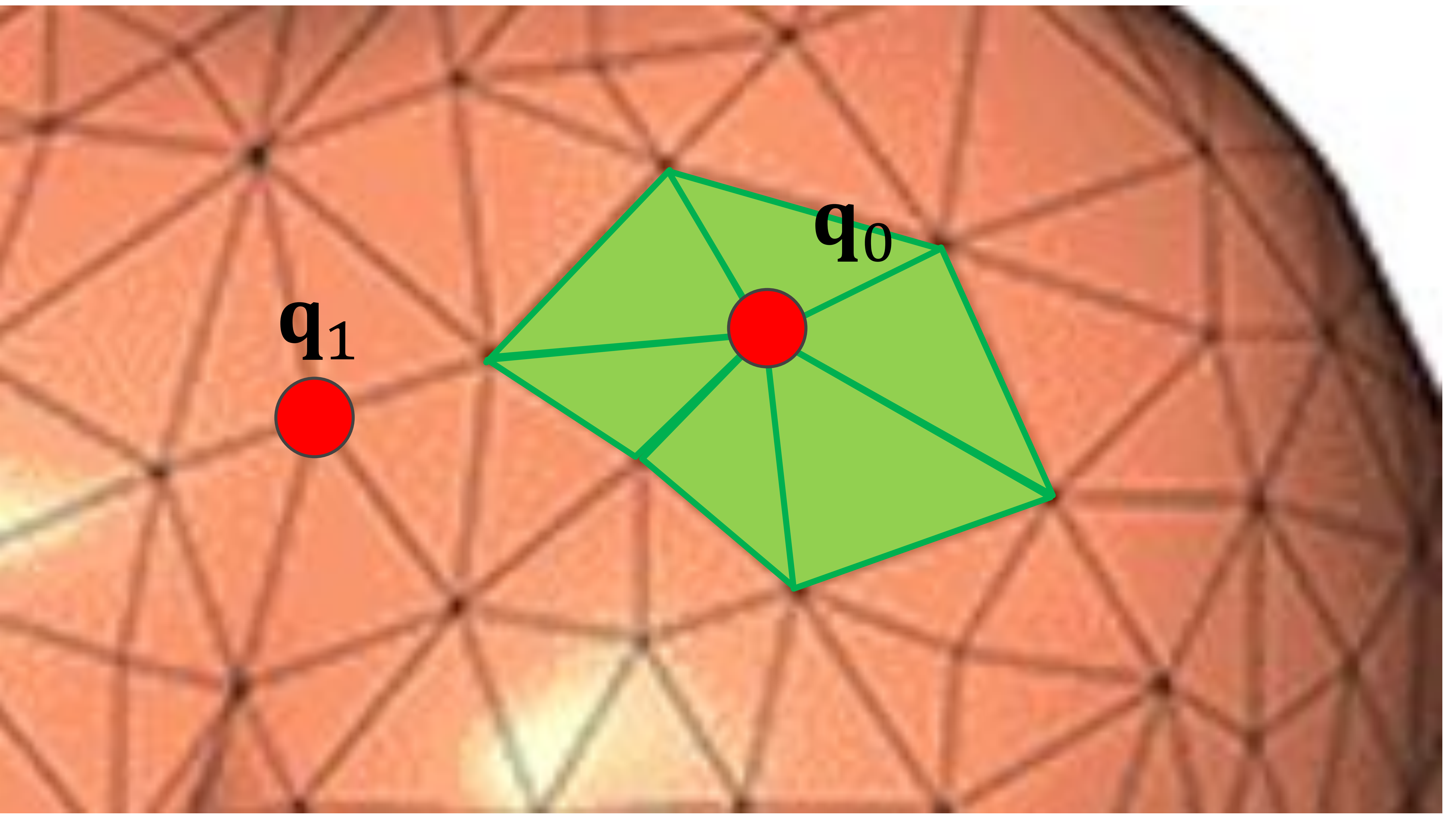
$$E\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right) = \frac{1}{2}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right)^T M_{\mathrm{jj}}\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + \mathbf{p}_j{}^T\left(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}\right) + h^2 V_j\left(\mathbf{q}_j{}^{\mathbf{i+1}}\right)$$

It’s not obvious but we can do this in parallel

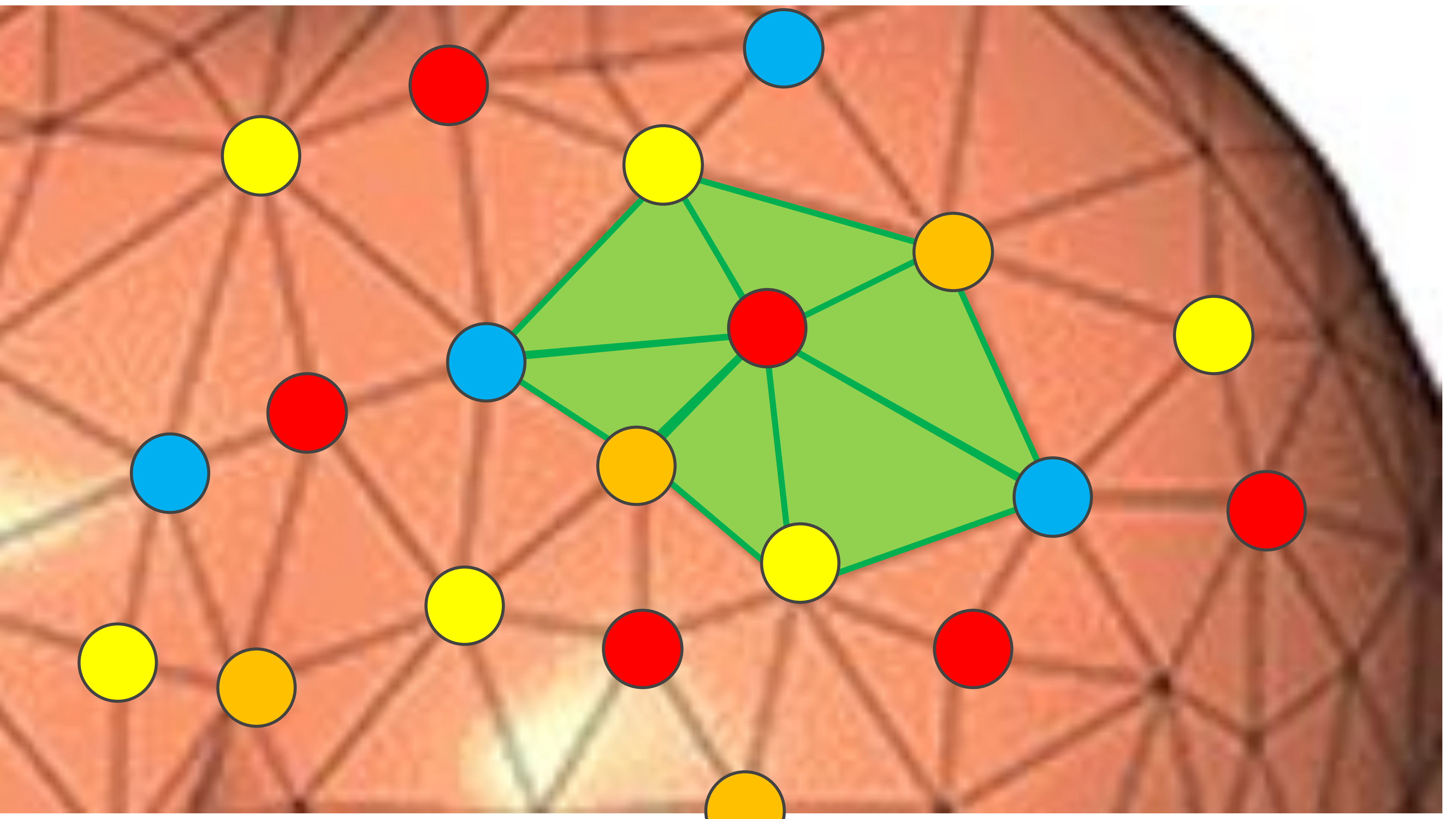


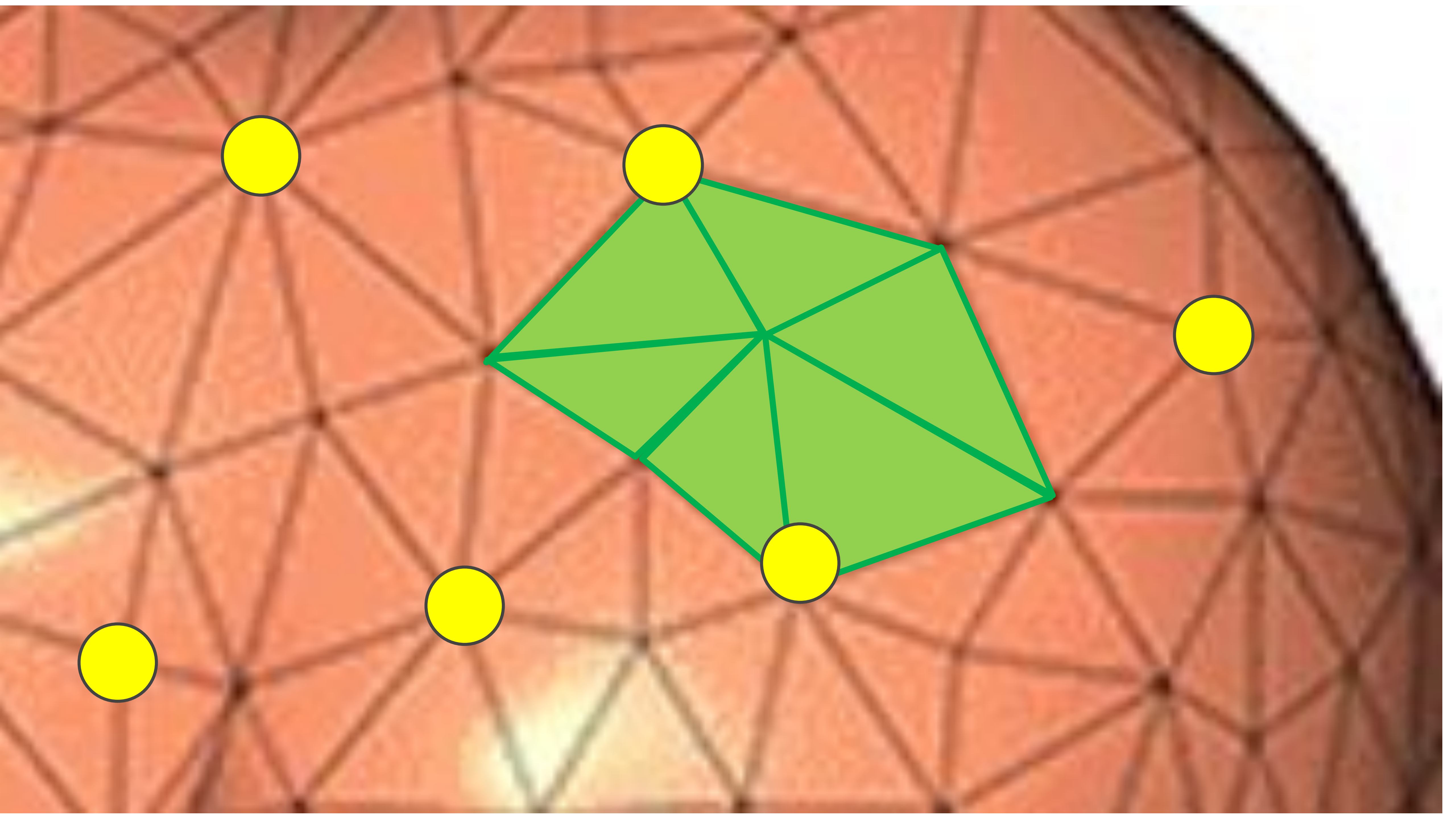
$\mathbf{q}_1$
$\mathbf{q}_0$


$\mathbf{q}_1$
$\mathbf{q}_0$



$\mathbf{q}_0$
$\mathbf{q}_1$



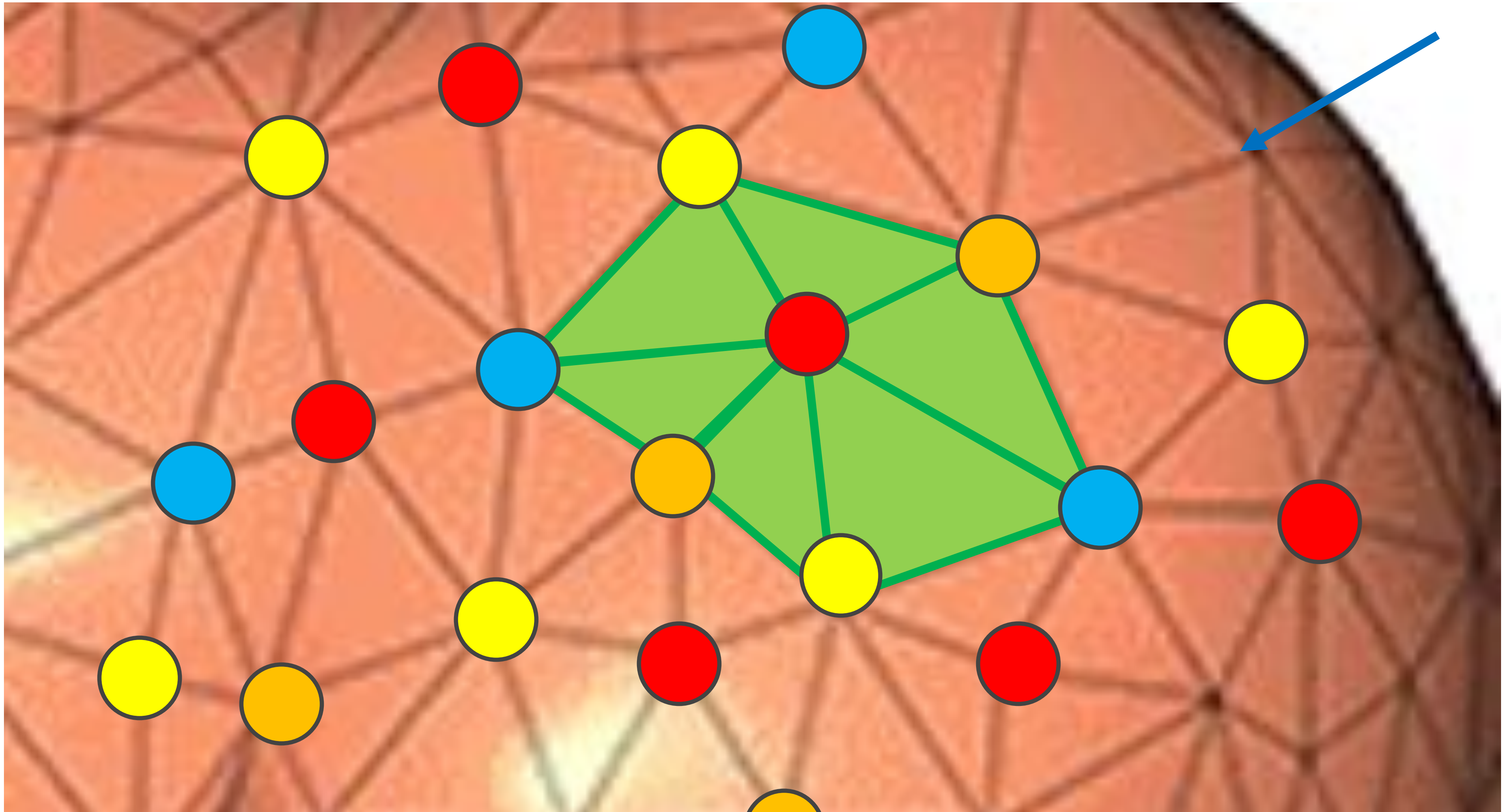


Greedy Graph Coloring

Pick a set of colors

Order all vertices in the mesh

Assign vertex first feasible color

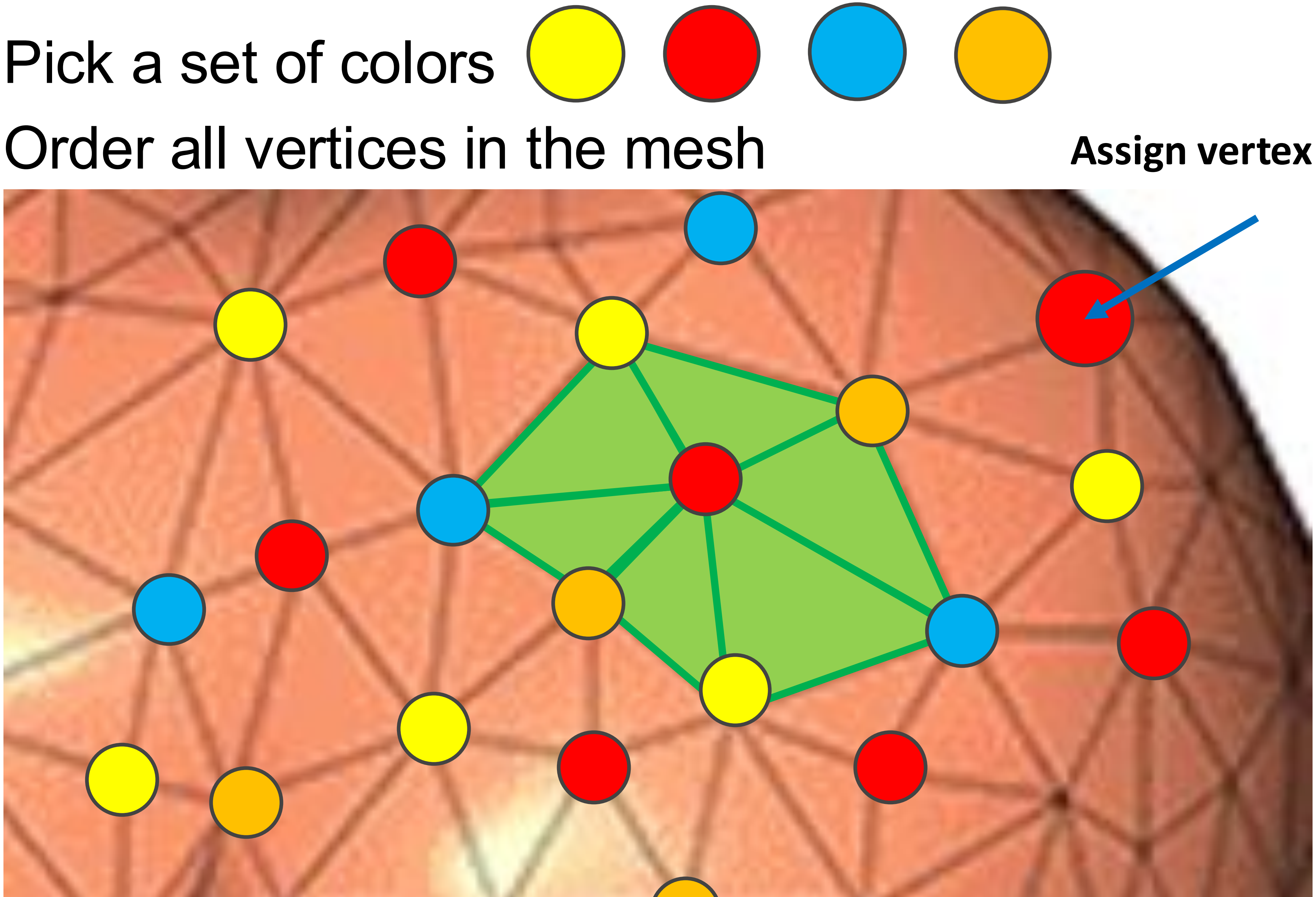


Greedy Graph Coloring

Pick a set of colors

Order all vertices in the mesh

Assign vertex first feasible color



Parallelization

Repeat for awhile

For each color,

In parallel, minimize for all $j \in$ color

$$E(\mathbf{q}_j^{\mathbf{i+1}}) = \frac{1}{2}(\mathbf{q}_j^{\mathbf{i+1}} - \tilde{\mathbf{q}}_j^{i})^T M_{jj}(\mathbf{q}_j^{\mathbf{i+1}} - \tilde{\mathbf{q}}_j^{i}) + \mathbf{p}_j^T(\mathbf{q}_j^{\mathbf{i+1}} - \tilde{\mathbf{q}}_j^{i}) + h^2 V_j(\mathbf{q}_j^{\mathbf{i+1}})$$


36M verts, 124M tets
avg frame time: 7.2s
max: 7.8s
Vertex Block Descent|Chen et a

Pros and Cons

Pros

Very, very fast due to parallelized small solves

Cons

None ?

Pros and Cons

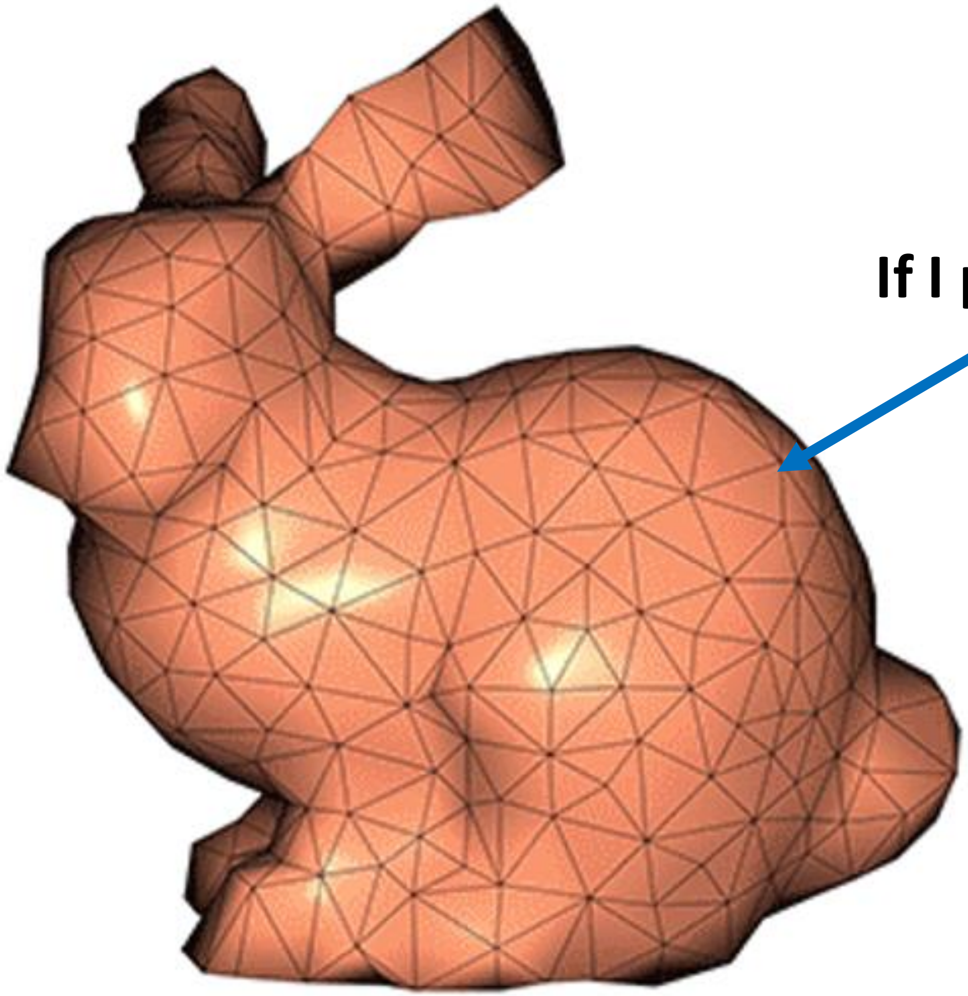
Pros

Very, very fast due to parallelized small solves

Cons

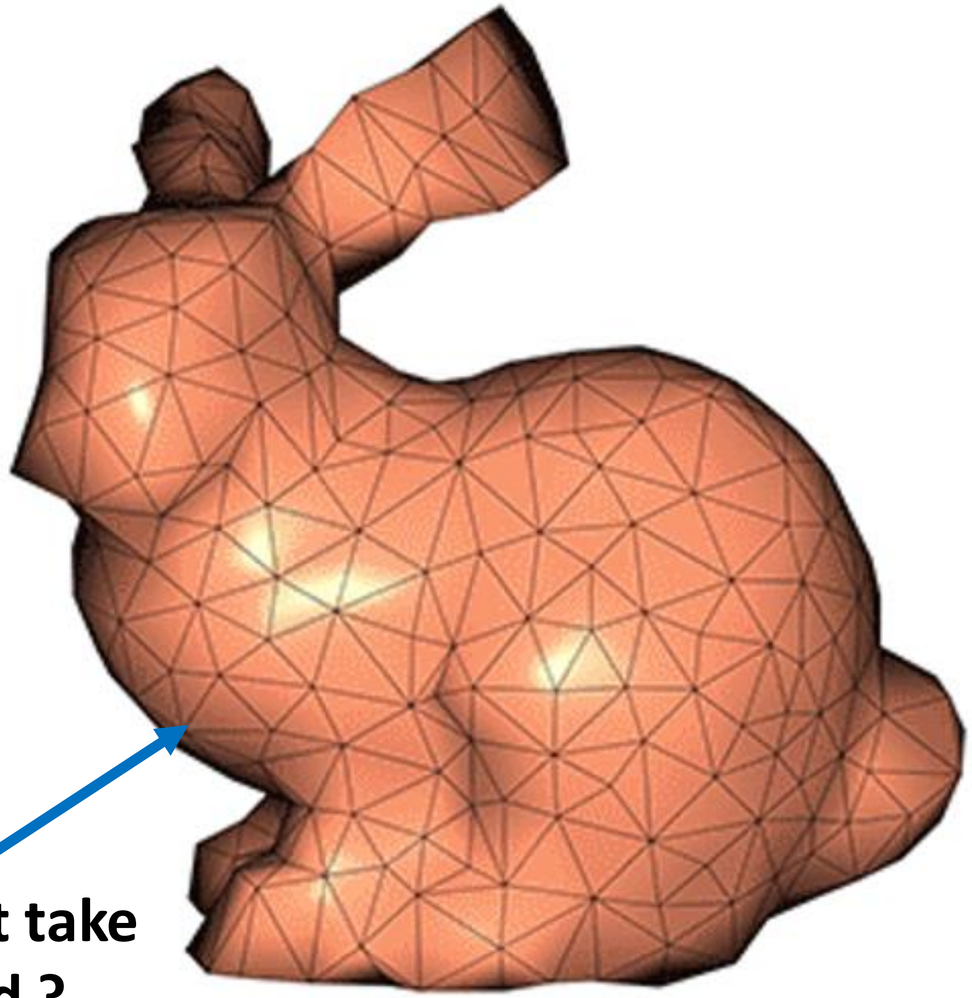
None ? (Sadly know)

Information Propagation



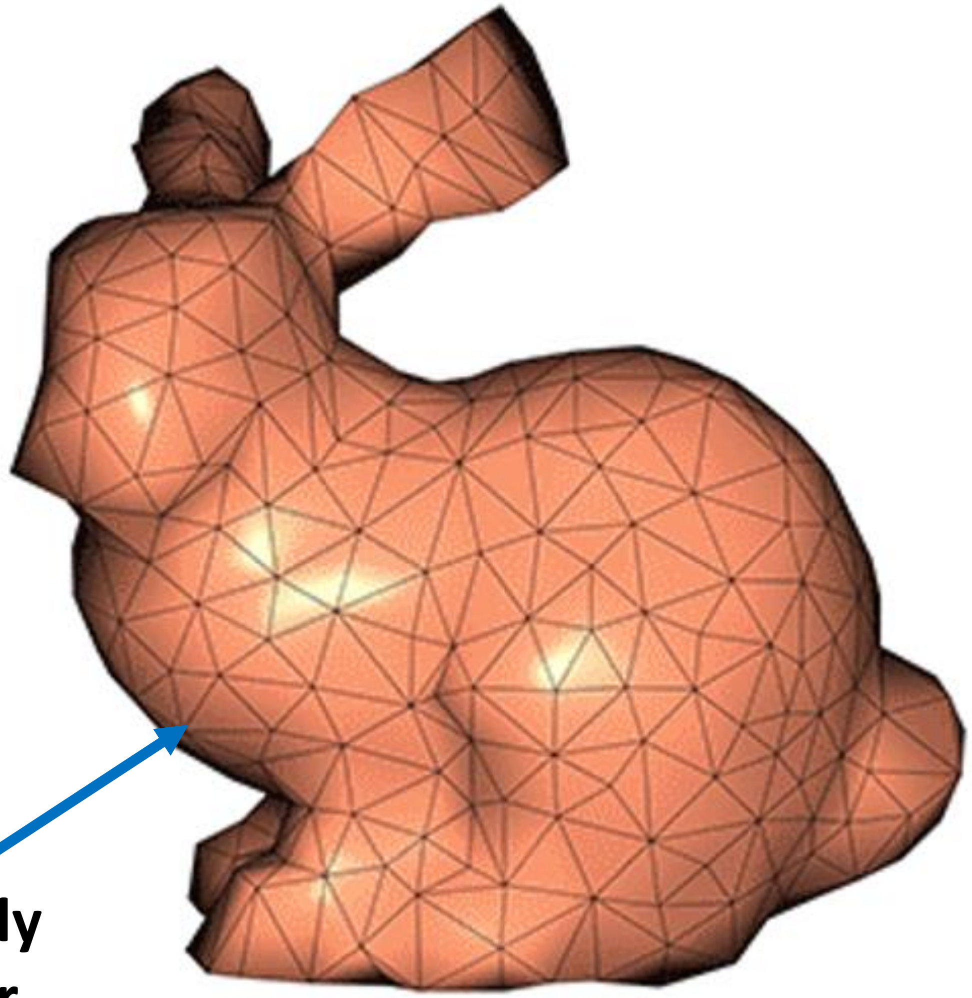
If I pull on this vertex

Information Propagation



How many iterations does it take for this vertex to respond ?

Information Propagation



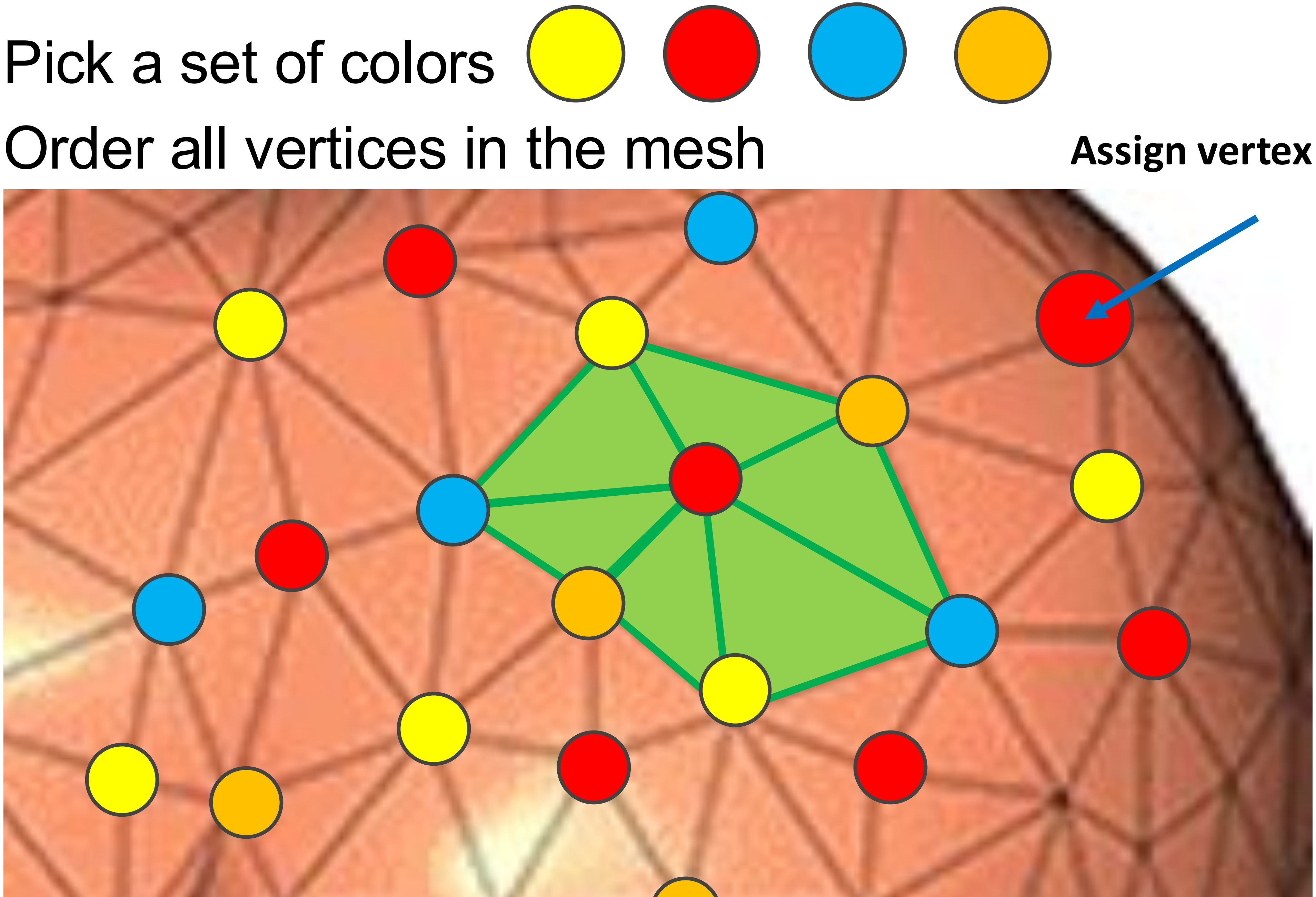
A lot ! The deformation only travels along one edge per iteration

Greedy Graph Coloring

Pick a set of colors

Order all vertices in the mesh

Assign vertex first feasible color



Parallelization

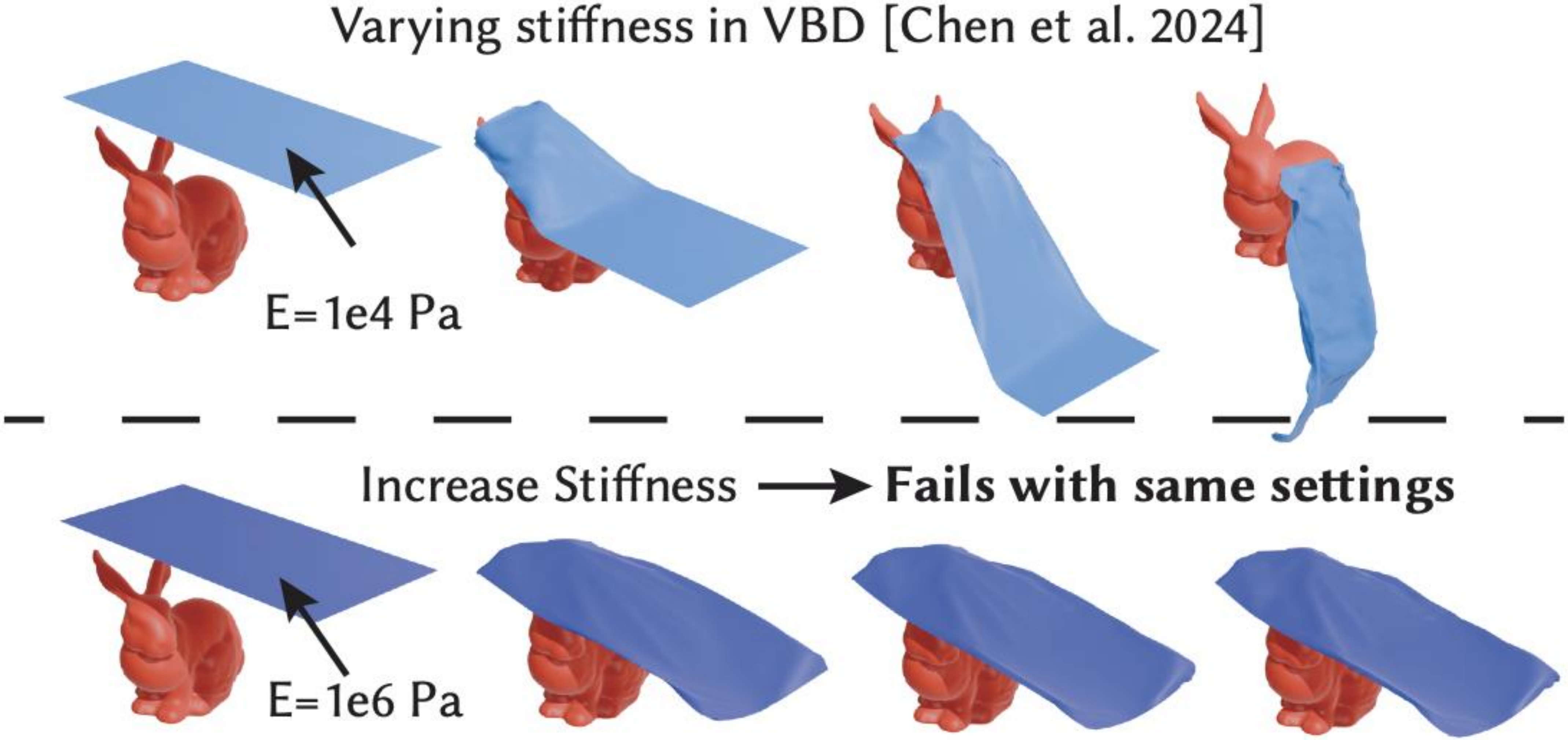
Repeat for awhile

For each color,

In parallel, minimize for all $j \in$ color

$$E(\mathbf{q}_j{}^{\mathbf{i+1}}) = \frac{1}{2}(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}})^T M_{\mathrm{jj}}(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}) + \mathbf{p}_j{}^T(\mathbf{q}_{\mathrm{j}}{}^{\mathbf{i+1}} - \tilde{\mathbf{q}}_{\mathrm{j}}^{\mathrm{i}}) + h^2 V_j(\mathbf{q}_j{}^{\mathbf{i+1}})$$

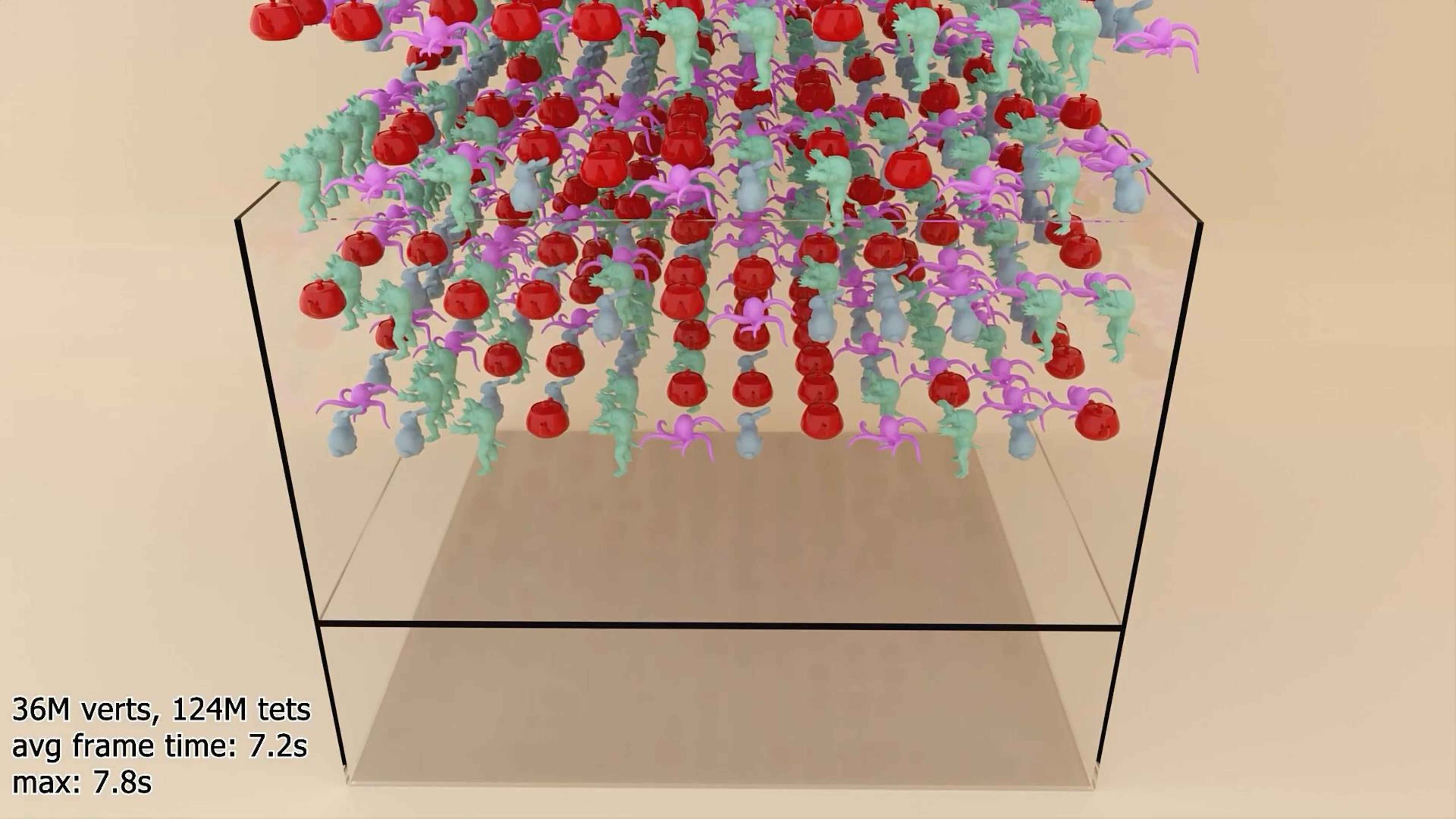
Information Propagation



Information Propagation

High resolution geometry and high stiffness exacerbates the issue

Not a deal breaker because you can tune your parameters and geometry to get reasonable output, but not a black box an algorithm as pure Newton's method.


36M verts, 124M tets
avg frame time: 7.2s
max: 7.8s

Next Lecture: NO MORE STRUCTURED LECTURES