Multi-Layer Thick Shells: Supplemental Document

Yunuo Chen, Tianyi Xie, Cem Yuksel, Danny Kaufman,
Yin Yang, Chenfanfu Jiang, Minchen Li

Contents

1 Derivation of Prism Elements

1.1 Deformation Gradient and Its Derivatives

For our isoparametric prism element, we derived that the deformation gradient at an arbitrary point $\mathbf{X}$ in the prism is

$$\mathbf{F}(\mathbf{q}) = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}(\mathbf{q}) = \frac{\partial \mathbf{x}}{\partial \mathbf{q}}(\frac{\partial \mathbf{X}}{\partial \mathbf{q}})^{-1}, \tag{1}$$

where

$$\frac{\partial \mathbf{X}}{\partial \mathbf{q}} = [\mathbf{X}_2 - \mathbf{X}_1 + \gamma(\mathbf{N}_2 - \mathbf{N}_1),\ \mathbf{X}_3 - \mathbf{X}_1 + \gamma(\mathbf{N}_3 - \mathbf{N}_1),\ \mathbf{N}_1 + \lambda_1(\mathbf{N}_2 - \mathbf{N}_1) + \lambda_2(\mathbf{N}_3 - \mathbf{N}_1)] \tag{2}$$

and

$$\frac{\partial \mathbf{x}}{\partial \mathbf{q}} = [\mathbf{x}_2 - \mathbf{x}_1 + \gamma(\mathbf{n}_2 - \mathbf{n}_1),\ \mathbf{x}_3 - \mathbf{x}_1 + \gamma(\mathbf{n}_3 - \mathbf{n}_1),\ \mathbf{n}_1 + \lambda_1(\mathbf{n}_2 - \mathbf{n}_1) + \lambda_2(\mathbf{n}_3 - \mathbf{n}_1)]. \tag{3}$$

Here we see that, unlike the linear tetrahedral elements, the deformation gradient is not constant in each prism element.

Let $\mathbf{B} = (\frac{\partial \mathbf{X}}{\partial \mathbf{q}})^{-1} \in \mathbb{R}^3$, we can derive the derivative of $\mathbf{F}$ w.r.t. the degrees of freedom $\mathbf{x}$ and $\mathbf{n}$ as

$$\frac{\partial\ \mathrm{vec}(\mathbf{F})}{\partial[\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}}, \mathbf{x}_3^{\mathrm{T}}, \mathbf{n}_1^{\mathrm{T}}, \mathbf{n}_2^{\mathrm{T}}, \mathbf{n}_3^{\mathrm{T}}]} = \Big(\begin{bmatrix} -\mathbf{I} & \mathbf{I} & \mathbf{0} & -\gamma\mathbf{I} & \gamma\mathbf{I} & \mathbf{0} \\ -\mathbf{I} & \mathbf{0} & \mathbf{I} & -\gamma\mathbf{I} & \mathbf{0} & \gamma\mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & (1-\lambda_1-\lambda_2)\mathbf{I} & \lambda_1\mathbf{I} & \lambda_2\mathbf{I} \end{bmatrix} \odot \mathbf{B} \Big) \in \mathbb{R}^{9\times 18}, \tag{4}$$

where we define the operator $\odot$ as: taking each column of the left operand, reshaping it to a 3×3 matrix in column major, premultiplying the reshaped matrix to $\mathbf{B}$, reshaping the resulting 3×3 matrix back to a

column vector in column major, and finally organizing all resulting column vectors together. In this way, we obtain

$$\frac{\partial \text{ vec}(\mathbf{F})}{\partial[\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}}, \mathbf{x}_3^{\mathrm{T}}]} = \begin{bmatrix} (-B_{11} - B_{21})\mathbf{I} & B_{11}\mathbf{I} & B_{21}\mathbf{I} \\ (-B_{12} - B_{22})\mathbf{I} & B_{12}\mathbf{I} & B_{22}\mathbf{I} \\ (-B_{13} - B_{23})\mathbf{I} & B_{13}\mathbf{I} & B_{23}\mathbf{I} \end{bmatrix} \tag{5}$$

and

$$\frac{\partial \text{ vec}(\mathbf{F})}{\partial[\mathbf{n}_1^{\mathrm{T}}, \mathbf{n}_2^{\mathrm{T}}, \mathbf{n}_3^{\mathrm{T}}]} = \begin{bmatrix} (-\gamma(B_{11} + B_{21}) + (1 - \lambda_1 - \lambda_2)B_{31})\mathbf{I} & (\gamma B_{11} + \lambda_1 B_{31})\mathbf{I} & (\gamma B_{21} + \lambda_2 B_{31})\mathbf{I} \\ (-\gamma(B_{12} + B_{22}) + (1 - \lambda_1 - \lambda_2)B_{32})\mathbf{I} & (\gamma B_{12} + \lambda_1 B_{32})\mathbf{I} & (\gamma B_{22} + \lambda_2 B_{32})\mathbf{I} \\ (-\gamma(B_{13} + B_{23}) + (1 - \lambda_1 - \lambda_2)B_{33})\mathbf{I} & (\gamma B_{13} + \lambda_1 B_{33})\mathbf{I} & (\gamma B_{23} + \lambda_2 B_{33})\mathbf{I} \end{bmatrix}, \tag{6}$$

which are useful in computing the nodal elasticity force and its Jacobian.

1.2 Consistent Mass Matrix

In FEM literature, if we define the shape function of node i as $N_i(\mathbf{X})$ and director i as $O_i(\mathbf{X})$ in an element, we have

$$\mathbf{x}(\mathbf{X}) = \sum_{i=0}^{3} \big(\mathbf{x_i} N_i(\mathbf{X}) + \mathbf{n_i} O_i(\mathbf{X})\big). \tag{7}$$

Based on our definition of

$$\mathbf{x}(\mathbf{q}) = \mathbf{x}_1 + \lambda_1(\mathbf{x}_2 - \mathbf{x}_1) + \lambda_2(\mathbf{x}_3 - \mathbf{x}_1) + \gamma(\mathbf{n}_1 + \lambda_1(\mathbf{n}_2 - \mathbf{n}_1) + \lambda_2(\mathbf{n}_3 - \mathbf{n}_1)), \tag{8}$$

we know that

$$\begin{aligned} &N_1(\mathbf{X}) = 1 - \lambda_1 - \lambda_2, \quad N_2(\mathbf{X}) = \lambda_1, \quad N_3(\mathbf{X}) = \lambda_2, \\ &O_1(\mathbf{X}) = \gamma(1 - \lambda_1 - \lambda_2), \quad O_2(\mathbf{X}) = \gamma\lambda_1, \quad O_3(\mathbf{X}) = \gamma\lambda_2. \end{aligned} \tag{9}$$

If we stack these shape functions together as $\mathbf{O}(\mathbf{X}) = [N_1(\mathbf{X}), N_2(\mathbf{X}), N_3(\mathbf{X}), O_1(\mathbf{X}), O_2(\mathbf{X}), O_3(\mathbf{X})]^{\mathrm{T}}$, we can write the consistent mass matrix of a prism element as

$$\mathbf{M} = \Big(\int_{\Omega_e^0} \rho_0 \mathbf{O}(\mathbf{X})\mathbf{O}(\mathbf{X})^{\mathrm{T}} \mathrm{d}\mathbf{X} \Big) \in \mathbb{R}^{6\times 6} \tag{10}$$

according to the weak form derivation of FEM. Here ρ_0 is the density at rest configuration, and each entry of the mass matrix applies to all the 3 dimensions of the corresponding two nodal positions or directors. For example,

$$M_{11} = \rho_0 \int_0^1 \int_0^{1-\lambda_1} \int_{-1}^1 (1 - \lambda_1 - \lambda_2)^2 (g + h\gamma + i\gamma^2) \mathrm{d}\gamma \mathrm{d}\lambda_2 \mathrm{d}\lambda_1 \tag{11}$$

after changing the integration variables to the parameters $\mathbf{q}$, where

$$\begin{aligned} g &= (\mathbf{X}_{12} \times \mathbf{X}_{13}) \cdot \mathbf{N}_1 + (\mathbf{X}_{12} \times \mathbf{X}_{13}) \cdot \mathbf{N}_{12}\lambda_1 + (\mathbf{X}_{12} \times \mathbf{X}_{13}) \cdot \mathbf{N}_{13}\lambda_2, \\ h &= (\mathbf{X}_{12} \times \mathbf{N}_3 + \mathbf{X}_{13} \times \mathbf{N}_2) \cdot \mathbf{N}_1 + (\mathbf{X}_{13} \times \mathbf{N}_{12}) \cdot \mathbf{N}_{13}\lambda_2 + (\mathbf{X}_{13} \times \mathbf{N}_{12}) \cdot \mathbf{N}_{13}\lambda_2, \\ i &= (\mathbf{N}_2 \times \mathbf{N}_3) \cdot \mathbf{N}_1, \end{aligned} \tag{12}$$

and $\mathbf{X}_{ij} = \mathbf{X}_j - \mathbf{X}_i$. Evaluating the integral analytically or via quadrature rules we obtain

$$M_{11} = \frac{1}{90}\rho_0 (3(\mathbf{X}_{12} \times \mathbf{X}_{13}) \cdot (\mathbf{N}_3 + \mathbf{N}_2 + 3\mathbf{N}_1) + 5(\mathbf{N}_2 \times \mathbf{N}_3) \cdot \mathbf{N}_1). \tag{13}$$

Similarly,

$$
\begin{aligned}
M_{12} = M_{21} &= \frac{1}{180}\rho_0(3(\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(\mathbf{N}_3+2\mathbf{N}_2+2\mathbf{N}_1)+5(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{13} = M_{31} &= \frac{1}{180}\rho_0(3(\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(2\mathbf{N}_3+\mathbf{N}_2+2\mathbf{N}_1)+5(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{14} = M_{41} &= \frac{1}{18}\rho_0(\mathbf{X}_{12}\times\mathbf{N}_3+\mathbf{X}_{13}\times\mathbf{N}_2)\cdot\mathbf{N}_1,\\
M_{15} = M_{51} &= \frac{1}{180}\rho_0((4\mathbf{X}_{12}\times\mathbf{N}_3+5\mathbf{X}_{13}\times\mathbf{N}_2)\cdot\mathbf{N}_1+(\mathbf{X}_{12}\times\mathbf{N}_{13})\cdot\mathbf{N}_2),\\
M_{16} = M_{61} &= \frac{1}{180}\rho_0((5\mathbf{X}_{12}\times\mathbf{N}_3+4\mathbf{X}_{13}\times\mathbf{N}_2)\cdot\mathbf{N}_1+(\mathbf{X}_{13}\times\mathbf{N}_{12})\cdot\mathbf{N}_3).
\end{aligned}
\tag{14}
$$

$$
\begin{aligned}
M_{22} &= \frac{1}{90}\rho_0(3(\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(\mathbf{N}_3+3\mathbf{N}_2+\mathbf{N}_1)+5(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{23} = M_{32} &= \frac{1}{180}\rho_0(3(\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(2\mathbf{N}_3+2\mathbf{N}_2+\mathbf{N}_1)+5(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{24} = M_{42} &= M_{15},\\
M_{25} = M_{52} &= \frac{1}{90}\rho_0((3\mathbf{X}_{12}\times\mathbf{N}_3+5\mathbf{X}_{13}\times\mathbf{N}_2)\cdot\mathbf{N}_1+2(\mathbf{X}_{12}\times\mathbf{N}_{13})\cdot\mathbf{N}_2),\\
M_{26} = M_{62} &= \frac{1}{2}M_{14}.
\end{aligned}
\tag{15}
$$

$$
\begin{aligned}
M_{33} &= \frac{1}{90}\rho_0(3(\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(3\mathbf{N}_3+\mathbf{N}_2+\mathbf{N}_1)+5(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{34} &= M_{43} = M_{16},\\
M_{35} &= M_{53} = M_{26},\\
M_{36} = M_{63} &= \frac{1}{90}\rho_0((5\mathbf{X}_{12}\times\mathbf{N}_3+3\mathbf{X}_{13}\times\mathbf{N}_2)\cdot\mathbf{N}_1+2(\mathbf{X}_{13}\times\mathbf{N}_{12})\cdot\mathbf{N}_3).
\end{aligned}
\tag{16}
$$

$$
\begin{aligned}
M_{44} &= \frac{1}{90}\rho_0((\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(\mathbf{N}_3+\mathbf{N}_2+3\mathbf{N}_1)+3(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{45} = M_{54} &= \frac{1}{180}\rho_0((\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(\mathbf{N}_3+2\mathbf{N}_2+2\mathbf{N}_1)+3(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{46} = M_{64} &= \frac{1}{180}\rho_0((\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(2\mathbf{N}_3+\mathbf{N}_2+2\mathbf{N}_1)+3(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1).
\end{aligned}
\tag{17}
$$

$$
\begin{aligned}
M_{55} &= \frac{1}{90}\rho_0((\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(\mathbf{N}_3+3\mathbf{N}_2+\mathbf{N}_1)+3(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1),\\
M_{56} = M_{65} &= \frac{1}{180}\rho_0((\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(2\mathbf{N}_3+2\mathbf{N}_2+\mathbf{N}_1)+3(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1).
\end{aligned}
\tag{18}
$$

$$
M_{66} = \frac{1}{90}\rho_0((\mathbf{X}_{12}\times\mathbf{X}_{13})\cdot(3\mathbf{N}_3+\mathbf{N}_2+\mathbf{N}_1)+3(\mathbf{N}_2\times\mathbf{N}_3)\cdot\mathbf{N}_1). \tag{19}
$$

With consistent mass matrix, gravity from a single element can be calculated as

$$
\sum_{j=1}^{3} M_{ij}\mathbf{g} \tag{20}
$$

for $\mathbf{x}_i$, and

$$
\sum_{j=1}^{3} M_{(i+3)j}\mathbf{g} \tag{21}
$$

for $\mathbf{n}_i$, where $\mathbf{g}$ is the gravitational acceleration vector. We will skip the derivation.

1.3 Switching the Degrees of Freedom to Surface Nodes

For the convenience of derivation, we defined the degrees of freedom as the midsurface nodal positions $\mathbf{x}^{mid}$ and directors $\mathbf{n}$. Switching the degrees of freedom to the surface nodes $\mathbf{y}$, we get

$$\mathbf{y}_{2j-1} = \mathbf{x}_j^{mid} - \mathbf{n}_j, \quad \mathbf{y}_{2j} = \mathbf{x}_j^{mid} + \mathbf{n}_j, \tag{22}$$

which indicates that

$$\frac{\mathrm{d}[\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}}, \mathbf{x}_3^{\mathrm{T}}, \mathbf{n}_1^{\mathrm{T}}, \mathbf{n}_2^{\mathrm{T}}, \mathbf{n}_3^{\mathrm{T}}]^{\mathrm{T}}}{\mathrm{d}[\mathbf{y}_1^{\mathrm{T}}, \mathbf{y}_2^{\mathrm{T}}, \mathbf{y}_3^{\mathrm{T}}, \mathbf{y}_4^{\mathrm{T}}, \mathbf{y}_5^{\mathrm{T}}, \mathbf{y}_6^{\mathrm{T}}]} = \begin{bmatrix} \frac{1}{2}\mathbf{I} & \frac{1}{2}\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \frac{1}{2}\mathbf{I} & \frac{1}{2}\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{1}{2}\mathbf{I} & \frac{1}{2}\mathbf{I} \\ -\frac{1}{2}\mathbf{I} & \frac{1}{2}\mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\frac{1}{2}\mathbf{I} & \frac{1}{2}\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & -\frac{1}{2}\mathbf{I} & \frac{1}{2}\mathbf{I} \end{bmatrix}. \tag{23}$$

With this derivative, we can conveniently transform the forces and their Jacobian matrices from the original degrees of freedom to $\mathbf{y}$ using chain rule.

2 Derivation of Complementary Wrinkle Coupling

2.1 Change of Variable with Null-Space Basis

With Lagrangian mechanics, we obtain the following spatially discretized dynamical system for our coupled fine membrane $\mathbf{x}$ and thick shell $\mathbf{y}$:

$$\begin{cases} \dfrac{\mathrm{d}\mathbf{v}_x}{\mathrm{d}t} = \mathbf{M}_x^{-1}(-\nabla E_x(\mathbf{x}) + \mathbf{P}\boldsymbol{\lambda}), & (24) \\ \dfrac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v}_x, & (25) \\ \dfrac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} = \mathbf{M}_y^{-1}(-\nabla E_y(\mathbf{y}) - \mathbf{P}^{\mathrm{T}}\mathbf{P}\boldsymbol{\lambda}), & (26) \\ \dfrac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{v}_y, & (27) \\ \mathbf{P}^{\mathrm{T}}(\mathbf{x} - \mathbf{P}\mathbf{y}) = \mathbf{0}. & (28) \end{cases}$$

Let $\mathbf{x} = \mathbf{P}\mathbf{y} + \mathbf{Q}\mathbf{z}$ where the columns of $\mathbf{Q}$ are the null-space basis of $\mathbf{P}$ we defined and $\mathbf{P}^{\mathrm{T}}\mathbf{Q} = \mathbf{0}$, Eq. 28 holds by construction, and Eq. 24 and Eq. 25 can be rewritten as

$$\begin{cases} \mathbf{P}\dfrac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} + \mathbf{Q}\dfrac{\mathrm{d}\mathbf{v}_z}{\mathrm{d}t} = \mathbf{M}_x^{-1}(-\nabla E_x(\mathbf{P}\mathbf{y} + \mathbf{Q}\mathbf{z}) + \mathbf{P}\boldsymbol{\lambda}), & (29) \\ \mathbf{P}\dfrac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} + \mathbf{Q}\dfrac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} = \mathbf{P}\mathbf{v}_y + \mathbf{Q}\mathbf{v}_z. & (30) \end{cases}$$

Here, we can see that the position updates of $\mathbf{y}$ and $\mathbf{z}$ are decoupled as $\mathbf{P}$ and $\mathbf{Q}$ are orthogonal, which gives us

$$\frac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} = \mathbf{v}_z \tag{31}$$

in addition to the equation of $\mathbf{y}$ given by Eq. 27. However, the velocity update could not be decoupled, but we can certainly eliminate the Lagrange multiplier vector $\boldsymbol{\lambda}$.

Premultiplying $\mathbf{Q}^{\mathrm{T}}\mathbf{M}_x$ to both sides of Eq. 29, we get

$$\mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{P}\frac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} + \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\frac{\mathrm{d}\mathbf{v}_z}{\mathrm{d}t} = -\mathbf{Q}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y} + \mathbf{Q}\mathbf{z}). \tag{32}$$

Similarly, premultiplying $\mathbf{P}^{\mathrm{T}}\mathbf{M}_x$ to both sides of Eq. 29, we get

$$\mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{P}\frac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\frac{\mathrm{d}\mathbf{v}_z}{\mathrm{d}t} = -\mathbf{P}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y}+\mathbf{Q}\mathbf{z}) + \mathbf{P}^{\mathrm{T}}\mathbf{P}\boldsymbol{\lambda}. \tag{33}$$

Then combining Eq. 33 and Eq. 26, we can eliminate $\mathbf{P}^{\mathrm{T}}\mathbf{P}\boldsymbol{\lambda}$ and obtain

$$(\mathbf{M}_y + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{P})\frac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\frac{\mathrm{d}\mathbf{v}_z}{\mathrm{d}t} = -\mathbf{P}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y}+\mathbf{Q}\mathbf{z}) - \nabla E_y(\mathbf{y}). \tag{34}$$

Reorganizing Eq. 34, 32, 31, 27 together, we get an unconstrained system

$$\begin{cases} \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{P}\dfrac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} + \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\dfrac{\mathrm{d}\mathbf{v}_z}{\mathrm{d}t} = -\mathbf{Q}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y}+\mathbf{Q}\mathbf{z}), \\ \dfrac{\mathrm{d}\mathbf{z}}{\mathrm{d}t} = \mathbf{v}_z, \\ (\mathbf{M}_y + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{P})\dfrac{\mathrm{d}\mathbf{v}_y}{\mathrm{d}t} + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\dfrac{\mathrm{d}\mathbf{v}_z}{\mathrm{d}t} = -\mathbf{P}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y}+\mathbf{Q}\mathbf{z}) - \nabla E_y(\mathbf{y}), \\ \dfrac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{v}_y. \end{cases} \tag{35}$$

2.2 Temporal Discretization and Optimization Form

Next, if we apply implicit Euler to discretize Eq. 35 (other time integration schemes can also be used), approximating any $\mathrm{d}\mathbf{q}/\mathrm{d}t$ as $(\mathbf{q}^{n+1}-\mathbf{q}^n)/h$, where h is the time step size, and using quantities in the $(n+1)$-th time step for the right hand sides, we obtain the fully discretized system

$$\begin{cases} \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{P}\dfrac{\mathbf{v}_y^{n+1}-\mathbf{v}_y^n}{h} + \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\dfrac{\mathbf{v}_z^{n+1}-\mathbf{v}_z^n}{h} = -\mathbf{Q}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y}^{n+1}+\mathbf{Q}\mathbf{z}^{n+1}), \\ \dfrac{\mathbf{z}^{n+1}-\mathbf{z}^n}{h} = \mathbf{v}_z^{n+1}, \\ (\mathbf{M}_y + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{P})\dfrac{\mathbf{v}_y^{n+1}-\mathbf{v}_y^n}{h} + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q}\dfrac{\mathbf{v}_z^{n+1}-\mathbf{v}_z^n}{h} = -\mathbf{P}^{\mathrm{T}}\nabla E_x(\mathbf{P}\mathbf{y}^{n+1}+\mathbf{Q}\mathbf{z}^{n+1}) - \nabla E_y(\mathbf{y}^{n+1}), \\ \dfrac{\mathbf{y}^{n+1}-\mathbf{y}^n}{h} = \mathbf{v}_y^{n+1}. \end{cases} \tag{36}$$

Now, we can clearly see that if we denote $\mathbf{w} = [\mathbf{y}^{\mathrm{T}}, \mathbf{z}^{\mathrm{T}}]^{\mathrm{T}}$, the nonlinear system in Eq. 36 is equivalent to the optimization problem

$$\mathbf{w}^{n+1} = \arg\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}-\tilde{\mathbf{w}}^n\|_{\mathbf{M}_w} + h^2\Big(E_x([\mathbf{P},\mathbf{Q}]\mathbf{w}) + E_y([\mathbf{I},\mathbf{0}]\mathbf{w})\Big) \tag{37}$$

followed by velocity update $\mathbf{v}_w^{n+1} = (\mathbf{w}^{n+1}-\mathbf{w}^n)/h$, where $\tilde{\mathbf{w}}^n = \mathbf{w}^n + h\mathbf{v}_w^n$ and

$$\mathbf{M}_w = \begin{bmatrix} \mathbf{M}_y + \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{P} & \mathbf{P}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q} \\ \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{P} & \mathbf{Q}^{\mathrm{T}}\mathbf{M}_x\mathbf{Q} \end{bmatrix}. \tag{38}$$

2.3 Pseudocode for Generating the Basis Matrices

```
void get_P_Q_mat(int nx, int ny,
                 vector<Vector3i> P_ind, vector<Vector2d> P_weight,
                 Mat& P, Mat& Q)
{
    /* nx: # high-res points (including those who are also low-res points),
       ny: # low-res points,
       P_ind: interpolating low-res points of each high-res point,
       P_weight: barycentric weights. */

    // P
    Triplets Tri;
    for (int xI = 0; xI < nx; ++xI) {
        Vector3i indI = P_ind[xI];
        Vector2d weightI = P_weight[xI];
        T w0 = (1 - weightI[0] - weightI[1]), w1 = weightI[0], w2 = weightI[1];
        if (w0 != 0)
            for (int d = 0; d < 3; ++d)
                Tri.emplace_back(xI * 3 + d, indI[0] * 3 + d, w0);
        if (w1 != 0)
            for (int d = 0; d < 3; ++d)
                Tri.emplace_back(xI * 3 + d, indI[1] * 3 + d, w1);
        if (w2 != 0)
            for (int d = 0; d < 3; ++d)
                Tri.emplace_back(xI * 3 + d, indI[2] * 3 + d, w2);
    }
    P.resize(nx * 3, ny * 3);
    P.setFromTriplets(Tri.begin(), Tri.end());

    // Q
    Tri.clear();
    map<int, int> y_to_x; // index map from low-res to high-res
    int nz = 0; // # high-res points that is not also a low-res point
    for (int xI = 0; xI < nx; ++xI) {
        Vector3i indI = P_ind[xI];
        Vector2d weightI = P_weight[xI];
        T w0 = (1 - weightI[0] - weightI[1]), w1 = weightI[0], w2 = weightI[1];
        if (w1 != 0 || w2 != 0) { // xI is not a low-res point
            if (w0 != 0)
                for (int d = 0; d < 3; ++d)
                    Tri.emplace_back(y_to_x[indI[0]] * 3 + d, nz * 3 + d, -w0);
            if (w1 != 0)
                for (int d = 0; d < 3; ++d)
                    Tri.emplace_back(y_to_x[indI[1]] * 3 + d, nz * 3 + d, -w1);
            if (w2 != 0)
                for (int d = 0; d < 3; ++d)
                    Tri.emplace_back(y_to_x[indI[2]] * 3 + d, nz * 3 + d, -w2);
            for (int d = 0; d < 3; ++d)
                Tri.emplace_back(xI * 3 + d, nz * 3 + d, 1.0);
            nz++;
        } else { // xI is also a low-res point
            y_to_x[indI[0]] = xI;
        }
    }
    Q.resize(nx * 3, nz * 3);
    Q.setFromTriplets(Tri.begin(), Tri.end());
}
```