Supplementary Material for
Modeling Wireframe Meshes with Discrete Equivalence Classes

This supplementary material provides implementation details about solving the global mesh optimization (Section 7) and local mesh optimization (Section 6) in the paper. In particular, the local mesh optimization problem is a small-scale version of the global mesh optimization problem. Hence, we first present the implementation details of solving the global mesh optimization problem, and then briefly introduce how the solver is slightly customized to solve the local mesh optimization problem.

1 Global Mesh Optimization

After clustering vertices and edges using hierarchical clustering with two prescribed tolerances, a set of template vertices and template edges can be computed. During global optimization, our goal is to minimize the difference from each vertex and edge in the mesh $\mathbf{M}$ to their corresponding templates by optimizing the 3D positions $\{\mathbf{v}_i\}$ of the mesh vertices. The global mesh optimization problem consists of four energy terms:

$$E = \lambda_1 E_{\text{vertex}} + \lambda_2 E_{\text{edge}} + \lambda_3 E_{\text{shape}} + \lambda_4 E_{\text{feature}} \tag{1}$$

$$\text{s.t.} \quad \theta_i > 2\arctan\frac{w}{r}, \quad \forall \text{ vertex angle}$$
$$l_j > 2R, \quad \forall \text{ mesh edge}$$

where E_{vertex} and E_{edge} are the sum of intra-cluster variance in each vertex and edge cluster, respectively, E_{shape} measures the mesh deformation from the mesh $\mathbf{M}$ to the reference input mesh $\mathbf{P}$, and E_{feature} indicates the degree to which user-specified features are preserved in the mesh $\mathbf{M}$. As directly minimizing these non-linear energy terms at the same time is challenging and time-consuming, we approximate Equation 1 as a least squares equation inspired by [Bouaziz et al. 2012; Bouaziz et al. 2014].

Least Squares Reformulation We convert Equation 1 into a least squares problem as following:

$$\min_{\mathbf{x}} \sum_i \lambda_i \|\mathbf{A}_i\mathbf{x} - \mathbf{b}_i\|^2 \tag{2}$$

where λ_i is the weight of each energy term, $\mathbf{A}_i$ is a constraint matrix of the ith energy term and $\mathbf{b}_i$ is a target vector. For a mesh with n vertices, solving Equation 2 is equivalent to finding $\mathbf{x}$ to satisfy the following least squares equation:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{3}$$

$$\mathbf{A}^T = (\sqrt{\lambda_1}\mathbf{A}_{\text{vertex}}, \sqrt{\lambda_2}\mathbf{A}_{\text{edge}}, \sqrt{\lambda_3}\mathbf{A}_{\text{shape}}, \sqrt{\lambda_4}\mathbf{A}_{\text{feature}})$$
$$\mathbf{x} = (\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n)$$
$$\mathbf{b}^T = (\sqrt{\lambda_1}\mathbf{b}_{\text{vertex}}, \sqrt{\lambda_2}\mathbf{b}_{\text{edge}}, \sqrt{\lambda_3}\mathbf{b}_{\text{shape}}, \sqrt{\lambda_4}\mathbf{b}_{\text{feature}})$$

where $\mathbf{A}$ is a $p \times n$ geometry constraint matrix, $\mathbf{x}$ is a $n \times 1$ vector consisting of all the mesh vertices that we are solving for, and $\mathbf{b}$ is a $p \times 1$ target vector. Note that p is determined by how large or complex the energy terms are. Among these four energy terms, formulating E_{shape} or E_{feature} as a least squares problem is a common knowledge. In the following, we will clarify how to reformulate E_{vertex} and E_{edge} to a least squares problem like Equation 2.

$\mathbf{E}_{\text{vertex}}$ Recall that E_{vertex} is the sum of the distances from each vertex to its corresponding centroid vertex:

$$E_{\text{vertex}} = \sum_{k=1}^{K_v} \sum_i \left(D_v(v_{k,i},\ \bar{v}_k)\right)^2 \tag{4}$$

where $v_{k,i}$ is the ith vertex in the kth vertex cluster, and $\bar{v}_k$ is the centroid of the kth vertex cluster. Given a m-valence mesh vertex $v_{k,i}$ and its target template vertex $\bar{v}_k$, we first use singular value decomposition to find the optimal alignment rotation matrix $\mathbf{T}_{k,i}$ from centroid vertex $\bar{v}_k$ to the vertex $v_{k,i}$ [Liu et al. 2023]. Thus $D_v(v_{k,i},\ \bar{v}_k)$ can be formulated as the following least squares equation:

$$D_v(v_{k,i},\ \bar{v}_k) = \|\mathbf{A}_{k,i}v_{k,i} - \mathbf{b}_{k,i}\| \tag{5}$$

$$\mathbf{A}_{k,i} = \begin{pmatrix} \frac{-1}{d_1} & \frac{1}{d_1} & 0 & \cdots & 0 \\ \frac{-1}{d_2} & 0 & \frac{1}{d_2} & \cdots & 0 \\ & & \vdots & & \\ \frac{-1}{d_m} & & \cdots & & \frac{1}{d_m} \end{pmatrix}$$
$$\mathbf{b}_{k,i} = {\mathbf{T}_{k,i}}^{\mathrm{T}}\bar{v}_k$$

$$\text{s.t.} \quad \mathbf{T}_{k,j}\mathbf{T}_{k,j}^{\mathrm{T}} = \mathbf{I}$$

where $\mathbf{A}_{k,i}$ and $\mathbf{b}_{k,i}$ are a $m \times (m+1)$ geometry constraint matrix and a $m \times 1$ target vector for $v_{k,i}$, respectively, d_m is the distance from $\mathbf{v}_{k,i}$ to m-th neighbouring vertex $\mathbf{v}_{k,i}^m$, $v_{k,i} = (\mathbf{v}_{k,i}, \mathbf{v}_{k,i}^1, \mathbf{v}_{k,i}^2, ..., \mathbf{v}_{k,i}^m)^T$ is represented by its own vertex and its m one-ring neighbouring vertices, and $\bar{v}_k$ is an $m+1$ vector $(\bar{v}_{k,0}, \bar{v}_{k,1}, , ..., \bar{v}_{k,m})^T$. Please refer to Section 5 in our paper for the detailed explanations of d_m, $v_{k,i}$ and $\bar{v}_k$. In our global optimization problem, we insert some all-zero columns to each $\mathbf{A}_{k,i}$ to form a matrix $\mathbf{A}'_{k,i}$ with n columns. Then we concatenate a set of $\mathbf{A}'_{k,i}$ and $\mathbf{b}_{k,i}$ to form the geometry constraint matrix $\mathbf{A}_{\text{vertex}}$ and target vector $\mathbf{b}_{\text{vertex}}$ for E_{vertex}, respectively.

$\mathbf{E}_{\text{edge}}$ Recall that E_{edge} is the sum of the difference from each edge length to its corresponding target edge length:

$$E_{\text{edge}} = \sum_{l=1}^{K_e} \sum_j \left(D_e(e_{l,j},\ \bar{e}_l)\right)^2 \tag{6}$$

where $e_{l,j}$ is the jth edge in the lth edge cluster, and $\bar{e}_l$ is the centroid of the lth edge cluster. $D_e(e_{l,j},\ \bar{e}_l)$ can be formulated as the following least squares equation:

$$D_e(e_{l,j},\ \bar{e}_l) = \|\mathbf{A}_{l,j}e_{l,j} - \mathbf{b}_{l,j}\| \tag{7}$$

$$\mathbf{A}_{l,j} = \left(\tfrac{1}{\bar{e}_l}, \tfrac{-1}{\bar{e}_l}\right)$$
$$\mathbf{b}_{l,j} = (\mathbf{v}_{l,j,1} - \mathbf{v}_{l,j,2})/\|\mathbf{v}_{l,j,1} - \mathbf{v}_{l,j,2}\|_2$$

where $\mathbf{A}_{l,j}$ and $\mathbf{b}_{l,j}$ are a 2×1 geometry constraint matrix and a 2×1 target vector for $e_{l,j}$, respectively, and $e_{l,j} = (\mathbf{v}_{l,j,1}, \mathbf{v}_{l,j,2})^T$ is a 1×1 vector defined by two end vertices. Then we concatenate a set of $\mathbf{A}_{l,j}$ and $\mathbf{b}_{l,j}$ to form the geometry constraint matrix $\mathbf{A}_{\text{edge}}$ and target vector $\mathbf{b}_{\text{edge}}$ for E_{edge}, respectively, in the same way as for E_{vertex}.

Solver We use successive over-relaxation (SOR) [Guennebaud et al. 2010] in Eigen library to solve Equation 2. Since the least squares method is not designed for solving optimization problems with hard constraints, the two hard constraints (for fabrication) in Equation 1 are not included in the least squares problem. However, this reformulation may lead to few vertex angles and/or mesh edges that do not satisfy the fabrication constraints. In our experiments, we found that these vertex angles and mesh edges are very few and only appear in mesh models with complex curvature. Moreover, vertex angles and mesh edges that do not satisfy the constraints are mostly marginal cases (i.e., close to the threshold). To address this issue, we first identify these vertex angles and mesh edges in the optimized mesh $\mathbf{M}$. If we find any such angle or edge, we slightly adjust the position of the corresponding vertex using a sampling-based approach to make the angle or edge satisfy the constraint. In case this local perturbation introduces few new vertex clusters and/or edge clusters, we will try to resolve (i.e., merge) them in the next iteration of local mesh optimization.

2 Local Mesh Optimization

Recall that we formulate an optimization problem to search for a feasible 3D position $\mathbf{v}$ of the vertex v to decrease K_v^n (Case 1) :

$$E_{\text{local}} = \omega_1 \big(D_v(v,\ \bar{v}_t)\big)^2 \ +\ \omega_2 \big(\text{Dist}(v,\ \mathbf{P})\big)^2 \ +\ \omega_3 \sum_k \big(D_v(v_k,\ \bar{v}_k)\big)^2 + \omega_4 \sum_l \big(D_e(e_l,\ \bar{e}_l)\big)^2 \tag{8}$$

$$\text{s.t.} \quad \theta_i > 2 \arctan \frac{w}{r}, \quad \forall \text{ mesh vertex} \in \{v, N(v)\}$$
$$l_j > 2R, \quad \forall \text{ mesh edge} \in I(v)$$

Similar to global mesh optimization, we reformulate Equation 8 to a least squares problem and solve it in the same way as the global mesh optimization. Comparing to the solver used in the global mesh optimization, the only difference is that we increase the number of samples when we slightly adjust the position $\mathbf{v}$ of optimized v. There are two reasons. First, vertex v should be reassigned to a new cluster which needs larger modification than requiring v stay in the same cluster. Second, the searching space is a 3D position $\mathbf{v}$ only so that the time complexity is still manageable after increasing the number of samples. We use the same approach to solve the local mesh optimization problem in the other two cases.

References

BOUAZIZ, S., DEUSS, M., SCHWARTZBURG, Y., WEISE, T., AND PAULY, M. 2012. Shape-Up: Shaping discrete geometry with projections. *Comp. Graph. Forum (SGP) 31*, 5, 1657–1667.

BOUAZIZ, S., MARTIN, S., LIU, T., KAVAN, L., AND PAULY, M. 2014. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. on Graph. (SIGGRAPH) 33*, 4, 154:1–154:11.

GUENNEBAUD, G., JACOB, B., ET AL., 2010. Eigen v3. url-http://eigen.tuxfamily.org.

LIU, Y., LEE, T.-U., KORONAKI, A., PIETRONI, N., AND XIE, Y. M. 2023. Reducing the number of different nodes in space frame structures through clustering and optimization. *Engineering Structures 284*, 116016:1–116016:10.