

On Mesh Movement by Spring Analogy

Hiroaki Nishikawa

October 2000

1 Iteration Formulas

1.1 An Iteration Formula

Consider the following iteration formula to reposition the node j .

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \frac{\sum_{\{T_j\}} w_T (\mathbf{x}_G^n - \mathbf{x}_j^n)}{\sum_{\{T_j\}} w_T} \quad (1)$$

where $\{T_j\}$ is a group of triangles that share a node j , the subscript G denotes the centroid of the triangle, and w_T is a constant weight defined within each triangle in $\{T_j\}$. The new position \mathbf{x}_j^{n+1} is therefore a weighted average of the centroids of the surrounding triangles. Or, in view of spring analogy, the node j is connected to the centroid of each surrounding triangle by a spring with stiffness w_T . At a new position, the force exerted to the node j by a spring associated with triangle T is then, by Hook's law,

$$F_T = w_T (\mathbf{x}_G^n - \mathbf{x}_j^{n+1}). \quad (2)$$

The node is in equilibrium when there vanishes the net force at j .

$$\sum_{\{T_j\}} F_T = \sum_{\{T_j\}} w_T (\mathbf{x}_G^n - \mathbf{x}_j^{n+1}) = 0 \quad (3)$$

which can be solved for the new position \mathbf{x}_j^{n+1}

$$\mathbf{x}_j^{n+1} = \frac{\sum_{\{T_j\}} w_T \mathbf{x}_G^n}{\sum_{\{T_j\}} w_T}, \quad (4)$$

which is equivalent to (1).

1.2 Minimization Problem

It is interesting that this formula can be derived from a minimization problem. Consider minimizing

$$\mathcal{F} = \frac{1}{6} \sum_{\{T\}} w_T \sum_{\{e_T\}} \ell_e^2 \quad (5)$$

where $\{T\}$ is a set of all the triangles in a domain, $\{e_T\}$ is a set of sides of triangle T , and ℓ_e is the length of edge e . Hence we minimize a weighted sum of the length of the edges. Taking the derivative with respect to a node j , we obtain

$$\frac{\partial \mathcal{F}}{\partial \mathbf{x}_j} = \sum_{\{T_j\}} w_T (\mathbf{x}_j - \mathbf{x}_G). \quad (6)$$

Then, using a steepest descent method, we have

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n - \omega \frac{\partial \mathcal{F}}{\partial \mathbf{x}_j} = \omega \sum_{\{T_j\}} w_T (\mathbf{x}_G - \mathbf{x}_j) \quad (7)$$

where ω is a small constant. A better formula may be constructed by using the second derivative,

$$\frac{\partial^2 \mathcal{F}}{\partial \mathbf{x}_j^2} = \mathbf{I} \sum_{\{T_j\}} w_T. \quad (8)$$

It is a Newton's method.

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n - \omega \frac{\partial \mathcal{F}}{\partial \mathbf{x}_j} \left[\frac{\partial^2 \mathcal{F}}{\partial \mathbf{x}_j^2} \right]^{-1} = \mathbf{x}_j^n + \omega \frac{\sum_{\{T_j\}} w_T (\mathbf{x}_G^n - \mathbf{x}_j^n)}{\sum_{\{T_j\}} w_T} \quad (9)$$

which is identical to (1) with the choice $\omega = 1$. This interpretation will be useful when the mesh movement is incorporated with diagonal swappings because they can be driven by the minimization of the functional.

1.3 Vertex Springs

There are other rearrangements possible for the formula (1). Firstly, we show that it is equivalent to the vertex spring analogy where the edges are considered as springs. Denote the nodes of a triangle in $\{T_j\}$ by j - R - L , so that

$$\mathbf{x}_G = \frac{1}{3}(\mathbf{x}_j + \mathbf{x}_R + \mathbf{x}_L). \quad (10)$$

Substituting this into (1)

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \frac{\sum_{\{T_j\}} \frac{w_T}{3} (\mathbf{x}_j^n + \mathbf{x}_R^n + \mathbf{x}_L^n - 3\mathbf{x}_j^n)}{\sum_{\{T_j\}} w_T}, \quad (11)$$

rearranging,

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \frac{\sum_{\{T_j\}} \frac{w_T}{3} \{(\mathbf{x}_R^n - \mathbf{x}_j^n) + (\mathbf{x}_L^n - \mathbf{x}_j^n)\}}{\sum_{\{T_j\}} w_T}, \quad (12)$$

finally,

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \frac{2}{3} \frac{\sum_{\{k_j\}} \alpha_{kj} (\mathbf{x}_k^n - \mathbf{x}_j^n)}{\sum_{\{k_j\}} \alpha_{kj}} \quad (13)$$

where $\{k_j\}$ is a set of nodes directly connected to j and

$$\alpha_{kj} = \frac{1}{2}(w_{kR} + w_{kL}). \quad (14)$$

Therefore the formula is equivalent, within a constant factor, to the one derived from the vertex spring analogy with the stiffness given by the average of the element stiffness between the two triangles that share that edge. Note however that it is not possible to determine w_T uniquely for given α_{kj} .

1.4 Segment Springs

Another alternative form is a so-called segment spring analogy where the analogy is applied to the displacements rather than the positions.

$$\delta \mathbf{x}_j^n = \delta \mathbf{x}_j^{n-1} + \frac{\sum_{\{k_j\}} \alpha_{kj} (\delta \mathbf{x}_k^{n-1} - \delta \mathbf{x}_j^{n-1})}{\sum_{\{k_j\}} \alpha_{kj}} \quad (15)$$

where $\delta \mathbf{x}_j$ is the displacement of the node j defined by

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \delta \mathbf{x}_j^n. \quad (16)$$

This can be shown to be equivalent to the vertex spring analogy,

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \frac{\sum_{\{k_j\}} \alpha_{kj} (\mathbf{x}_k^n - \mathbf{x}_j^n)}{\sum_{\{k_j\}} \alpha_{kj}}. \quad (17)$$

Add and subtract the necessary quantities to get

$$\delta \mathbf{x}_j^n = \delta \mathbf{x}_j^{n-1} - \mathbf{x}_j^n + \mathbf{x}_j^{n-1} + \frac{\sum_{\{k_j\}} \alpha_{kj} (\mathbf{x}_k^n - \mathbf{x}_j^n)}{\sum_{\{k_j\}} \alpha_{kj}} \quad (18)$$

which can be written as

$$\delta \mathbf{x}_j^n = \delta \mathbf{x}_j^{n-1} + \frac{\sum_{\{k_j\}} \alpha_{kj} (\mathbf{x}_k^n - \mathbf{x}_j^n - \mathbf{x}_j^n + \mathbf{x}_j^{n-1})}{\sum_{\{k_j\}} \alpha_{kj}}, \quad (19)$$

and rearranged

$$\delta \mathbf{x}_j^n = \delta \mathbf{x}_j^{n-1} + \frac{\sum_{\{k_j\}} \alpha_{kj} \{(\mathbf{x}_k^n - \mathbf{x}_k^{n-1}) + (\mathbf{x}_k^{n-1} - \mathbf{x}_j^n) - (\mathbf{x}_j^n - \mathbf{x}_j^{n-1})\}}{\sum_{\{k_j\}} \alpha_{kj}} \quad (20)$$

where the term in the middle on the numerator vanishes by (17), and therefore we obtain

$$\delta \mathbf{x}_j^n = \delta \mathbf{x}_j^{n-1} + \frac{\sum_{\{k_j\}} \alpha_{kj} (\delta \mathbf{x}_k^{n-1} - \delta \mathbf{x}_j^{n-1})}{\sum_{\{k_j\}} \alpha_{kj}} \quad (21)$$

which is the desired result. Note that this algorithm is more expensive than others because it is necessary to store the displacements as additional variables at nodes.

1.5 Remarks

Firstly, we remark that none of the above algorithms achieve precise equidistribution of anything. All they do is deform the mesh into some equilibrium state. Secondly, none of the algorithm guarantee that the mesh remains valid during the iterations. It can happen that some element volumes go negative, especially the group of triangles $\{T_j\}$ form a nonconvex polygon.

2 Equidistribution

We consider equidistributing a quantity in the form $Q_e \Delta s_e^2$ associated with an edge e by minimizing

$$\mathcal{F} = \frac{1}{4} \sum_{\{e\}} (Q_e \Delta s_e^2 - C)^2 \quad (22)$$

where C is the average value of $Q_e \Delta s_e$ over the edges. Suppose Q_e are constants (or frozen), then we obtain the update formula.

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n - \omega \frac{\partial \mathcal{F}}{\partial \mathbf{x}_j} \left[\frac{\partial^2 \mathcal{F}}{\partial \mathbf{x}_j^2} \right]^{-1} \quad (23)$$

where

$$\frac{\partial \mathcal{F}}{\partial \mathbf{x}_j} = \sum_{\{e_j\}} \alpha_e (Q_e \Delta s_e^2 - C) (\mathbf{x}_j^n - \mathbf{x}_k^n) \quad (24)$$

$$\left[\frac{\partial^2 \mathcal{F}}{\partial \mathbf{x}_j^2} \right] = \sum_{\{e_j\}} \{ Q_e (Q_e \Delta s_e^2 - C) \mathbf{I} + 2Q_e^2 (\mathbf{x}_j^n - \mathbf{x}_k^n)^2 \} \quad (25)$$

Or maybe we can simply take

$$\left[\frac{\partial^2 \mathcal{F}}{\partial \mathbf{x}_j^2} \right] = \sum_{\{e_j\}} Q_e (Q_e \Delta s_e^2 - C) \mathbf{I} \quad (26)$$

Then,

$$\mathbf{x}_j^{n+1} = \mathbf{x}_j^n + \frac{\sum_{\{k_j\}} \alpha_{kj} (\mathbf{x}_k^n - \mathbf{x}_j^n)}{\sum_{\{k_j\}} \alpha_{kj}}. \quad (27)$$

where

$$\alpha_{kj} = Q_e (Q_e \Delta s_e^2 - C). \quad (28)$$

This shows that the node j moves towards the node with which it shares a large Q_e , or it moves such that $Q_e \Delta s_e^2$ is equally distributed by adjusting the edge length Δs_e .