OPT-*i*
An International Conference on
Engineering and Applied Sciences Optimization
M. Papadrakakis, M.G. Karlaftis, N.D. Lagaros (eds.)
Kos Island, Greece, 4-6, June 2014

# RIGID MOTION MESH MORPHER: A NOVEL APPROACH FOR MESH DEFORMATION

**George S. Eleftheriou**[1,2], **Guillaume Pierrot**[1]

[1]ESI-Group
99 rue des Solets, Parc d'Affaires SILIC, 94513 Rungis CEDEX, France
[2]National Technical University of Athens
P.O. Box 64069, 15710 Athens, Greece
e-mail: {george.eleftheriou,guillaume.pierrot}@esi-group.com

**Keywords:** rigid motion mesh morpher, as-rigid-as-possible, mesh morphing, mesh deformation, mesh anisotropy, mesh rotation, distortion, mesh-less, optimization-based, cfd, computational fluid dynamics, adjoint based optimization

**Abstract.** *In adjoint based shape optimization problems, after the sensitivities have been computed, there are two ways of dealing with the necessary changes to the mesh. The first one is re-meshing based on the new shape, while the second one is adapting the existing mesh (by moving the nodes) to fit the new shape. Re-meshing may be time consuming, as it is introduced as a separate step inside the optimization loop, and tedious as it may require by-hand manipulation. It also introduces inconsistencies in the process as the sensitivities have been computed at isoconnectivity. Morphing also has its share of challenges, namely maintaining the mesh quality (avoiding distorted and negative cells) while deforming it. Within this context, various mesh morphing techniques have been developed. The Spring Analogy [1] is simple but may suffer robustness issues. Laplacian smoothing [11] is suitable for translation but does not account for rotation. The linear elasticity approach [3] does not account for mesh anisotropy and is difficult to implement for general meshes because finite elements are used to solve the equations. Finally, the Radial Basis Functions [7] are promising but computationally heavy as the matrices involved are full, restricting the mesh size and complicating the implementation. The Rigid Motion Mesh Morpher approach, proposed in the present study, aims at overcoming the above-mentioned limitations, being more flexible and essentially mesh-less, since it does not require any inertial quantities or cell connectivities related to the mesh. Firstly, the set of boundary nodes (nodes defining the shape) of the mesh is identified. The prescribed motion of these nodes (their velocities) is known. Then, all nodes are grouped into "stencils" which are required to deform in an as-rigid-as-possible way. Hence, every stencil has a rotation velocity and a translation velocity. Those, along with the velocities of all internal nodes, form the set of unknowns. The as-close-to-rigid-as-possible motion is ensured by attempting to minimize a metric representing the difference of the actual deformation from a perfectly rigid motion (a translation plus a rotation). It will be shown that this quantity is related to the anisotropic deformation energy. Next, the resulting system of equations is solved, having the prescribed motion of the boundary nodes as boundary conditions.*

# 1 INTRODUCTION

Let us consider the general shape optimization problem which, in gradient-based optimization problems, can be described as

$$\min_{\vec{d}\in\mathcal{D}} J(U,\vec{d}) \tag{1}$$

$$R(U,\vec{d}) = 0 \tag{2}$$

With

- $J$: the objective function
- $U$: the system state given by the natural problem $R$ (Navier-Stokes equations) defined on the domain $\Omega$ (if steady) or $\Omega \times (0,T]$ (if unsteady)
- $\vec{d} \in \mathcal{D} \subset \mathbb{R}^D, D \geq 1$ the design variables of the model. In our study this vector is the nodes coordinates of the mesh $M_h$ related to the physical domain $\Omega$. Let us denote by $\{N_i\}_{i=1}^N$ the set of these nodes and by $\Omega_B$ and $\Omega_I$ respectively the boundary and the interior of the domain $\Omega$. $I_I = \{i/\ N_i \in \Omega_I\}$ and $I_B = \{i/\ N_i \in \Omega_B\}$ are the sets of node indices belonging respectively to $\Omega_I$ and $\Omega_B$.

In the case of a gradient-based method, the $-\nabla_{\vec{d}} J$ at each optimization iteration gives the best descent direction. The computed gradient concerns the boundary nodes and the interior nodes must somehow follow their movement. This can be achieved by trashing the existing mesh and creating a new one (re-meshing) based on the new shape. However it is time-consuming and introduces inconsistencies, as the connectivities of the sequence of meshes produced differ from iteration to iteration. It also requires by-hand manipulation.

Another approach is to just make the internal nodes follow the movement of the boundary nodes defining the shape, thereby deforming the existing mesh. The cell connectivities are kept intact during the process and there is no new mesh to be built. This is called mesh morphing or mesh smoothing. The problem of this approach is that the movement does not necessarily end up well for the resulting mesh quality. It might be severly distorted, having negative cells.

Within this context, several mesh morphing methods have been developed. They are classified in four principal categories described in the subsections 1.1, 1.2, 1.3 and 1.4. We attempt to quickly cite the principles of the techniques along with their weaknesses and merits. To facilitate notation we can globally define as $\vec{g} = (\vec{u}_i)_{i\in I_B}$ the displacement prescribed over $\Omega_B$.

## 1.1 Spring analogy

The spring analogy method [4, 2, 12, 1] is the most simple and classical approach, performing quite well in a number of cases. Unfortunately, it doesn't prevent cell collapse, and fails when the local mesh motion exceeds significantly the local mesh size. Things become even worse, when it has to deal with mesh anisotropy and mesh rotation. Its principal idea is the introduction of a spring energy $E_s$ and the requirement for it to be minimum

$$E_s = \sum_{ij} \frac{1}{L_{ij}} (u_j - u_i)^2 \tag{3}$$

where $L_{ij}$ is the length of the spring.

### 1.2 Laplacian smoothing

Also known as "Laplacian coordinates" or "Laplacian based", this approach is based on the solution of a simple Laplace equation [10, 13, 11, 6] such that the displacement $\vec{u}$ of the nodes' coordinates is the solution of the following problem:

$$\begin{cases} \nabla \cdot (\mu(\vec{x}) \nabla \vec{u}) = 0 & \text{in } \Omega \\ \vec{u} = \vec{g} & \text{on } \Omega_B \end{cases} \tag{4}$$

with $\mu(\vec{x})$ as the diffusion coefficient selected over $\Omega$ preserving the mesh anisotropy. The computation of the displacement $\vec{u}$ related to the internal nodes (4) comes down to solving a linear system whose dimension is equal to the number of nodes. This method seems efficient when the mesh is subject to translation but suffers robustness issues in cases of more complex transformations such as rotation.

### 1.3 Linear elasticity

In this method [3, 10, 5], in order to handle rotation, a linear elasticity model is introduced to compute the displacements $\vec{u}$ such that:

$$\begin{cases} \nabla \cdot \sigma(\vec{u}) = 0 & \text{in } \Omega \\ \vec{u} = \vec{g} & \text{on } \Omega_B \end{cases} \tag{5}$$

$\sigma$ being the Cauchy stress tensor

$$\begin{cases} \sigma & = & \lambda \, tr(\varepsilon(\vec{u}) \mathbb{I} + 2\mu\varepsilon(\vec{u}) \\ \varepsilon(\vec{u}) & = & \frac{1}{2}(\vec{u}^T + \vec{u}) \end{cases} \tag{6}$$

with $\lambda$ and $\mu$ being the Lamé constants. This method reaches its limits with large rotations or mesh anisotropy as it doesn't handle them intrinsically. Furthermore it is difficult to implement for general polyhedral meshes because finite elements are used to solve the equations.

### 1.4 Radial Basis Function interpolation

This is an interpolation method [8, 7] searching for the displacement $\vec{u}(\vec{x})$

$$\vec{u}(\vec{x}) = \sum_l \gamma_l \, \phi(\|\vec{x} - \vec{x}_l\|) + h(\vec{x}) \tag{7}$$

$\vec{u}$ is the smooth interpolant which minimizes a certain norm. $\phi$ is the radial basis function [7]. The constraint is to be exact over the boundary nodes:

$$\vec{u}(\vec{x_i}) = \vec{\mathbf{v}}(\vec{x_i}) \quad i \in I_B. \tag{8}$$

The search for $\{\gamma_l\}_{l=1}^{N_B}$ satisfying (8), requires the solution of a linear system, the matrix of which is unfortunately dense. This causes difficulties in cases of large $N_B$.

In order to determine the coefficients $\{\gamma_l\}_{l=1}^{\#I_B}$ we have to solve a linear system in $\mathbb{R}^{(N_B+1)\times(N_B+1)}$ where $N_B = \#I_B$. Compared to the previously cited techniques, RBF morphing proves quite robust with respect to both mesh rotation and mesh anisotropy. The problem is that the matrices involved in the computations are always dense by design. Therefore, it's a matter of choice between a simple implementation at high computational cost, or a complex implementation at the cost of solving the system almost as if it was sparse.

## 2 THE RIGID MOTION MESH MORPHER CONTRIBUTION

The method belongs to the family of ***optimization-based*** methods according to the classification cited in [9]. This means that the nodes are not moved based on a heuristic algorithm such as Laplacian smoothing, they are moved to minimize a given distortion metric. More specifically, it tries to favour rigidity in the critical directions of imminent distortion.

There is an inherent contradiction starting from the very name of this method. Rigid Motion Mesh Morpher. Why Rigid Motion? Afterall, how could something (a mesh in particular) be kept rigid while being deformed (morphed)? The stated objective of this method and its main philosophy is to deform a mesh, meanwhile keeping elements/parts of it as-rigid-as-possible. Obviously they cannnot be kept entirely rigid, as this would defeat the purpose, however there are ways to favour rigidity in certain critical directions when distortion of the element/part becomes imminent. These elements/parts are essentially groups of nodes called ***stencils***. During the mesh deformation, there is a kind of prevention mechanism that kicks-in whenever a particular stencil is about to distort or degenerate to a line segment. The stencils can be defined with freedom as to the choice of nodes belonging to them. In the simple case, a 1:1 ratio between the number of stencils and nodes could be considered as follows; a stencil may consist of one node, plus all of its adjacent/neighbouring nodes sharing a cell. The main qualities of the method is that it is mesh-less and handles intrinsically mesh anisotropy and mesh rotation.

Let us assume that an arbitrary rigid stencil $s$ is in motion. Its motion comes as the combination of a translation, under velocity $\vec{a_s}$ and a rotation under velocity $\vec{b_s}$. In this case, the velocity of a node $j$ belonging to it, denoted as $\vec{v_{sj}}$ would be

$$\vec{v_{sj}} = \vec{a_s} + \vec{b_s} \wedge (\vec{x_j} - \vec{c_s}) \tag{9}$$

where $\vec{x_j}$ is the coordinate vector of the node and $\vec{c_s}$ is the coordinate vector of the stencil's center of gravity. In the simple case, its coordinates can be computed as the mean value of the coordinates of all nodes constituting the stencil (center of gravity of a point cloud). Moving further, we no longer require that the stencil $s$ be rigid; it is now allowed to deform. Therefore the velocity of the node, $\vec{v_j}$ would no longer be equal to $\vec{v_{sj}}$

$$\vec{v_j} \neq \vec{a_s} + \vec{b_s} \wedge (\vec{x_j} - \vec{c_s}) = \vec{v_{sj}} \tag{10}$$

Let $\vec{e_{sj}}$ be a vector representing this difference, between the actual deformation of the stencil and its perfectly rigid motion. This vector will be a function of the stencil $s$ and the nodes $j$ belonging to it:

$$\vec{e_{sj}} = \vec{v_j} - \vec{v_{sj}} = \vec{v_j} - \vec{a_s} - \vec{b_s} \wedge (\vec{x_j} - \vec{c_s}) \tag{11}$$

Taking control over how this vector evolves/develops for every single stencil of the mesh and trying to bound/minimize it, to the extent that this is possible, gives the first idea of how the optimization problem will be posed. Therefore we enhance/enrich it with some weighting coefficients

$$\vec{e_{sj}} = \sqrt{w_s \mu_{sj}}(\vec{v_j} - \vec{a_s} - \vec{b_s} \wedge (\vec{x_j} - \vec{c_s})) \tag{12}$$

The square root is there just for convenience and the need for it will become clearer in section 5.3; $w_s$ is a scalar weight per stencil that may arbitrarily stress the importance of some stencils as higher than others'. It can be objective when set to $1.0$. To better explain the need for a per stencil-node scalar weight $\mu_{sj}$ though, an example will be provided. Consider, for instance,

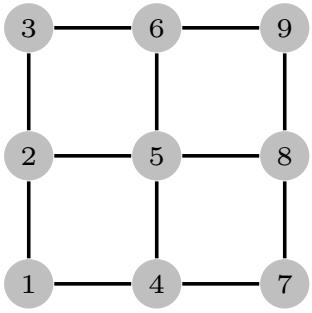
an anisotropic stencil, depicted in figure 1b, that is in a critical condition, meaning that if it is allowed to further deform and squeeze, it risks degenerating to a line segment or distort.

This sounds like exactly the case where it would be useful to favour rigidity in the direction of squeeze, freezing any further deformation. This is where $\mu_{sj}$ comes in handy serving as a limiter to further distortion. For instance, it could be something like

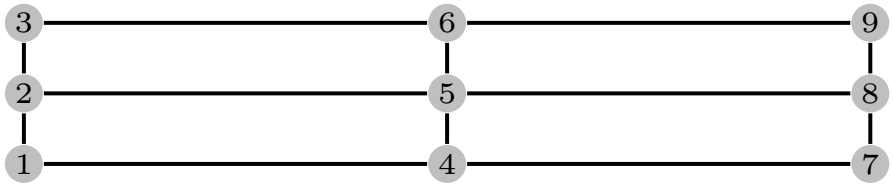
$$\mu_{sj} = \frac{\exp\left(-\frac{||\vec{x_j} - \vec{c_s}||^2}{h^2}\right)}{||\vec{x_j} - \vec{c_s}||^2 + \varepsilon} \tag{13}$$

$h$ being the average size of the mesh and $\varepsilon$ an arbitrary positive constant just to avoid division by zero when the node under consideration coincides with the stencil's center of gravity. The coefficient and its underlying logic will be detailed in section 4.

The quantity $\vec{e_{sj}}^T \cdot \vec{e_{sj}}$ (scalar), henceforth denoted as $\|\vec{e_{sj}}\|^2$, will be shown (later) that relates to the anisotropic deformation energy of the stencil.




(a) Isotropic stencil




(b) Squeezed/anisotropic stencil (we favour rigidity in the direction of squeeze)

Figure 1: Isotropic and anisotropic stencils

## 3 TOTAL DISTORTION ENERGY

At this point, a quantity called ***"total distortion energy"*** of the mesh will be introduced, to serve as the objective function and at the same time the distortion metric, bound to be minimized by the morphing algorithm. It is

$$E(\vec{u_j}, \vec{a_s}, \vec{b_s}) = \sum_s \sum_{j \in s} \|\vec{e_{sj}}\|^2 \tag{14}$$

which is quadratic. Hence, to find the stationary points of the total distortion energy and minimize it we require for all nodes $j$ and stencils $s$

$$\frac{\partial E}{\partial \vec{v_j}} = \frac{\partial E}{\partial \vec{a_s}} = \frac{\partial E}{\partial \vec{b_s}} = 0 \tag{15}$$

## 4 ASYMPTOTIC DEVELOPMENT

We will see in this section how to relate the distortion metric of a stencil $\mathcal{S}$:

$$\mathcal{E}_s = \sum_{j \in \mathcal{S}} \mu_{sj} \ \left\{ \vec{v}_j - \vec{\alpha}_s - \vec{\beta}_s \wedge (\vec{x}_j - \vec{c}_s) \right\}^2 \tag{16}$$

to the strain rate tensor:

$$\dot{\varepsilon} = \left\{ \frac{1}{2} \ \left( \frac{\partial v^\alpha}{\partial x_\beta} + \frac{\partial v^\beta}{\partial x_\alpha} \right) \right\}_{1 \leq \alpha \leq d}^{1 \leq \beta \leq d} \tag{17}$$

The first remark is that if $\left\{ (\vec{v}_i)_{i \in I_I}, \left( \vec{\alpha}_s, \vec{\beta}_s \right)_{s \in \mathbb{S}} \right\}$ is solution of the optimisation problem (15) then we surely have:

$$\left( \vec{\alpha}_s, \vec{\beta}_s \right) = \underset{\vec{a}_s, \vec{b}_s}{\operatorname{argmin}} \sum_{j \in \mathcal{S}} \mu_{sj} \ \left\{ \vec{v}_j - \vec{a}_s - \vec{b}_s \wedge (\vec{x}_j - \vec{c}_s) \right\}^2 \tag{18}$$

And subsequently:

$$\left( \sum_{j \in \mathcal{S}} \mu_{sj} \right) \vec{\alpha}_s = \sum_{j \in \mathcal{S}} \mu_{sj} \ \left\{ \vec{v}_j - \vec{\beta}_s \wedge (\vec{x}_j - \vec{c}_s) \right\} \tag{19}$$

and:

$$\mathbb{Y}_s \, \vec{\beta}_s = \sum_{j \in \mathcal{S}} \mu_{sj} \ (\vec{x}_j - \vec{c}_s) \wedge (\vec{v}_j - \vec{\alpha}_s) \tag{20}$$

with:

$$\mathbb{Y}_s = \left( \sum_{j \in \mathcal{S}} \mu_{sj} \ \{\vec{e}_\delta \wedge (\vec{x}_j - \vec{c}_s)\} \cdot \{\vec{e}_\gamma \wedge (\vec{x}_j - \vec{c}_s)\} \right)_{1 \leq \delta \leq d}^{1 \leq \gamma \leq d} \tag{21}$$

where $(\vec{e}_\delta)_{1 \leq \delta \leq d}$ denotes the canonical basis of $\mathbb{R}^d$.
Up to the change of variable:

$$\vec{\alpha}_s \leftarrow \vec{\alpha}_s + \vec{\beta}_s \wedge (\vec{x}_s - \vec{c}_s) \tag{22}$$

where:

$$x_s = \frac{\sum_{j \in \mathcal{S}} \mu_{sj} \, \vec{x}_j}{\sum_{j \in \mathcal{S}} \mu_{sj}} \tag{23}$$

equation (19) and (20) become:

$$\vec{\alpha}_s = \frac{\sum_{j \in \mathcal{S}} \mu_{sj} \, \vec{v}_j}{\sum_{j \in \mathcal{S}} \mu_{sj}} \tag{24}$$

and:

$$\tilde{\mathbb{Y}}_s \, \vec{\beta}_s = \sum_{j \in \mathcal{S}} \mu_{sj} \, (\vec{x}_j - \vec{x}_s) \wedge (\vec{v}_j - \vec{\alpha}_s) \tag{25}$$

with:

$$\tilde{\mathbb{Y}}_s = \left( \sum_{j \in \mathcal{S}} \mu_{sj} \ \{\vec{e}_\delta \wedge (\vec{x}_j - \vec{x}_s)\} \cdot \{\vec{e}_\gamma \wedge (\vec{x}_j - \vec{x}_s)\} \right)_{1 \leq \delta \leq d}^{1 \leq \gamma \leq d} \tag{26}$$

Let's now assume that $(\vec{v}_j)_{j \in \mathcal{S}}$ is the interpolated of a smooth vector field $\vec{v}$ at nodes location:

$$(\vec{v}_j)_{j \in \mathcal{S}} = (\vec{v}(\vec{x}_j))_{j \in \mathcal{S}}$$

We have the following Taylor expansion:

$$\vec{v}(\vec{x}_j) = \vec{v}(\vec{x}_s) + \mathcal{D}\vec{v} \ \ (\vec{x}_j - \vec{x}_s) + \mathcal{O}(h^2) \quad \forall j \in \mathcal{S} \tag{27}$$

where $h$ is a characteristic size of the stencil and:

$$\mathcal{D}\vec{v} = \left\{ \frac{\partial v^\alpha}{\partial x_\beta} \right\}_{1 \leq \alpha \leq d}^{1 \leq \beta \leq d} \tag{28}$$

is the vector field differential at $\vec{x}_s$.

By the definition of $\vec{x}_s$ (23) and (24) it comes then that:

$$\vec{\alpha}_s = \vec{v}(\vec{x}_s) + \mathcal{O}(h^2) \tag{29}$$

And (25) becomes:

$$\tilde{\mathbb{Y}}_s \, \vec{\beta}_s = \sum_{j \in \mathcal{S}} \mu_{sj} \, (\vec{x}_j - \vec{x}_s) \wedge (\mathcal{D}\vec{v} \;\; (\vec{x}_j - \vec{x}_s)) + \mathcal{O}(h^3) \tag{30}$$

Now, by introducing the classical splitting of $\mathcal{D}\vec{v}$ between its symmetric and anti-symmetric part:

$$\mathcal{D}\vec{v} = \dot{\varepsilon} + \dot{\Omega} \tag{31}$$

where:

$$\dot{\varepsilon} = \left\{ \frac{1}{2} \, \left( \frac{\partial v^\alpha}{\partial x_\beta} - \frac{\partial v^\beta}{\partial x_\alpha} \right) \right\}_{1 \leq \alpha \leq d}^{1 \leq \beta \leq d} \tag{32}$$

is the so-called rotation rate tensor, it comes:

$$\tilde{\mathbb{Y}}_s \, (\vec{\beta}_s - \vec{\omega}_s) = \mathcal{O}(h^3) \tag{33}$$

where

$$\vec{\omega}_s = \frac{1}{2} \vec{\nabla} \wedge \vec{v}$$

is the so-called rotation rate vector.

Hence we have:

$$\vec{\beta}_s = \vec{\omega}_s + \mathcal{O}(h) \tag{34}$$

Using now (27),(29) and (34), and substituting back in equation (16) it comes:

$$\begin{aligned} \mathcal{E}_s &= \sum_{j \in \mathcal{S}} \mu_{sj} \; \left\{ \vec{v}_j - \vec{v}(\vec{x}_s) - \vec{\omega}_s \wedge (\vec{x}_j - \vec{x}_s) + \mathcal{O}(h^2) \right\}^2 \\ &= \sum_{j \in \mathcal{S}} \mu_{sj} \; \left\{ \dot{\varepsilon}_s \, (\vec{x}_j - \vec{x}_s) + \mathcal{O}(h^2) \right\}^2 \\ &= (\mathbb{X}_s \, \dot{\varepsilon}_s) : \dot{\varepsilon}_s + \mathcal{O}(h^3) \end{aligned} \tag{35}$$

where : denotes the Frobenius product of 2 matrices and:

$$\mathbb{X}_s = \left( \sum_{j \in \mathcal{S}} \mu_{sj} \, (\vec{x}_j - \vec{x}_s)^\alpha \, (\vec{x}_j - \vec{x}_s)^\beta \right)_{1 \leq \alpha \leq d}^{1 \leq \beta \leq d} \tag{36}$$

is the so-called *stencil anisotropy tensor*. We will now consider 2 practical cases. The first one corresponds to a pure isotropic stencil (Fig. 1a).

With the weights $\mu_{sj}$ being chosen as indicated in (13), it is easy to see that:

$$\vec{c}_s = \vec{x}_s = \vec{x}_5$$

where $\vec{x}_5$ denotes the central node of the stencil and:

$$\mathbb{X}_s = \mu_s \, \mathcal{I}d$$

with:

$$\begin{aligned} \mu_s &= \left\{ \sum_{j \in \mathcal{S}} \exp\left( -\frac{\|\vec{x_j} - \vec{x_5}\|^2}{h^2} \right) \frac{(x_j^1 - x_5^1)^2}{||\vec{x_j} - \vec{x_5}||^2 + \varepsilon} \right\} \\ &= \left\{ \sum_{j \in \mathcal{S}} \exp\left( -\frac{\|\vec{x_j} - \vec{x_5}\|^2}{h^2} \right) \frac{(x_j^2 - x_5^2)^2}{||\vec{x_j} - \vec{x_5}||^2 + \varepsilon} \right\} \end{aligned}$$

Consequently, there is no favoured direction on the stencil distortion metric.

In case of a squeezed mesh, on the other hand (Fig. 1b), as it happens in boundary layers for instance, things are quite different.

Indeed, we have again that

$$\vec{c}_s = \vec{x}_s = \vec{x}_5$$

but the stencil anisotropy tensor is no longer scalar:

$$\mathbb{X}_s = \text{DIAG}\,(\mu_s^\alpha)_{1 \leq \alpha \leq d}$$

with:

$$\mu_s^\alpha = \left\{ \sum_{j \in \mathcal{S}} \exp\left( -\frac{\|\vec{x_j} - \vec{x_5}\|^2}{h^2} \right) \frac{\left(x_j^\alpha - x_5^\alpha\right)^2}{||\vec{x_j} - \vec{x_5}||^2 + \varepsilon} \right\}$$

Because this time the stencil dimension in the y direction $h_y$ is way smaller than in the x direction $h_x$ and we have that:

$$c(\mathbb{X}_s) = \frac{h_x}{h_y}$$

where $c(\mathbb{X}_s)$ denotes the condition number of the matrix. Consequently it comes:

$$\mathcal{E}_s = h_x \left\{ c(\mathbb{X}_s)\, \dot{\varepsilon}_{yy}^2 + \dot{\varepsilon}_{xx}^2 \right\} + \mathcal{O}(h^3)$$

and we see that we put more rigidity in the direction of squeezeness, which is is to be preferred as it is the direction of most imminent breakdown of the mesh. In this sense we can say that the Rigid Motion Mesh Morpher algorithm handles *naturally* mesh anistropy.

## 5 BUILDING THE SYSTEM OF EQUATIONS

The quadratic minimization problem of the total distortion energy of the mesh, as stated in (15), brings us to the following symmetric positive definite system

$$\begin{bmatrix} \mathbb{A}_{uu} & \mathbb{A}_{u(a|b)} \\ \mathbb{A}_{(a|b)u} & \mathbb{A}_{(a|b)(a|b)} \end{bmatrix} \cdot \begin{bmatrix} u \\ (a|b) \end{bmatrix} = \begin{bmatrix} P_u \\ P_{(a|b)} \end{bmatrix} \tag{37}$$

where the RHS consists of the boundary conditions, namely the prescribed nodes' velocities. Attempting to solve it using the Schur complement leads to two different cases of elimination:

$$\begin{aligned} \text{Either} \quad u = f(a|b) \quad \text{or} \\ (a|b) = g(u) \end{aligned} \tag{38}$$

So for instance

$$\begin{aligned} u = -\mathbb{A}_{uu}^{-1} \cdot \mathbb{A}_{u(a|b)} \cdot (a|b) + \mathbb{A}_{uu}^{-1} \cdot P_u \rightarrow \\ (-\mathbb{A}_{(a|b)u} \cdot \mathbb{A}_{uu}^{-1} \cdot \mathbb{A}_{u(a|b)} + \mathbb{A}_{(a|b)(a|b)}) \cdot (a|b) = -\mathbb{A}_{(a|b)} \cdot \mathbb{A}_{uu}^{-1} \cdot P_u + P_{(a|b)} \end{aligned} \tag{39}$$

Specializing the above analysis, for $\dfrac{\partial E}{\partial \vec{v_j}} = 0$ we have:

$$\frac{\partial E}{\partial \vec{v_j}} = \frac{\partial}{\partial \vec{v_j}} \left( \sum_s \sum_{j \in s} \|\vec{e_{sj}}\|^2 \right) = \frac{\partial}{\partial \vec{v_j}} \left( \sum_{j \in s_1} \|\vec{e_{s_1 j}}\|^2 + \sum_{j \in s_2} \|\vec{e_{s_2 j}}\|^2 + ... + \sum_{j \in s_N} \|\vec{e_{s_N j}}\|^2 \right) \tag{40}$$

So for a stencil:

$$\frac{\partial}{\partial \vec{v_j}} \sum_{j \in s_k} \|\vec{e_{s_k j}}\|^2 = \begin{cases} \displaystyle\sum_{j \in s_k} \frac{\partial \|\vec{e_{s_k j}}\|^2}{\partial \vec{v_j}} & \text{if } j \in s_k \\ 0 & \text{if } j \notin s_k \end{cases} \tag{41}$$

Considering the above, equation (40) becomes

$$\frac{\partial E}{\partial \vec{v_j}} = \sum_{s \ni j} \frac{\partial \|\vec{e_{sj}}\|^2}{\partial \vec{v_j}} = \sum_{s \ni j} 2 \vec{e_{sj}}^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} = 0 \tag{42}$$

But

$$\sum_{s \ni j} 2 \vec{e_{sj}}^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} = 0 \Leftrightarrow \sum_{s \ni j} \vec{e_{sj}}^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} = 0 \Leftrightarrow \sum_{s \ni j} \left[ \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \right]^T \vec{e_{sj}} = 0 \tag{43}$$

Since $\vec{e_{sj}}$ is linear with respect to $\vec{v_j}, \vec{a_s}, \vec{b_s}$, it could also be expressed as follows:

$$\vec{e_{sj}} = \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \vec{v_j} + \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} \vec{a_s} + \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \vec{b_s} \tag{44}$$

Hence

$$\frac{\partial E}{\partial \vec{v_j}} = \sum_{s \ni j} \left[ \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \right]^T \left( \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \vec{v_j} + \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} \vec{a_s} + \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \vec{b_s} \right) = 0 \tag{45}$$

We denote by

$$\mathbb{N}_j = \sum_{s \ni j} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \tag{46}$$

$$\mathbb{H}_{js} = \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}}\right]^T \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} \quad \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right] \tag{47}$$

the equation becomes

$$\mathbb{N}_j \vec{v_j} = -\sum_{s \ni j} \mathbb{H}_{js} \begin{bmatrix} \vec{a_s} \\ \vec{b_s} \end{bmatrix} \tag{48}$$

For $\dfrac{\partial E}{\partial \vec{a_s}} = 0$ we have

$$\frac{\partial E}{\partial \vec{a_s}} = \frac{\partial}{\partial \vec{a_s}} \left( \sum_s \sum_{j \in s} \|\vec{e_{sj}}\|^2 \right) = \frac{\partial}{\partial \vec{a_s}} \left( \sum_{j \in s_1} \|\vec{e_{s_1 j}}\|^2 + \sum_{j \in s_2} \|\vec{e_{s_2 j}}\|^2 + ... + \sum_{j \in s_N} \|\vec{e_{s_N j}}\|^2 \right) \tag{49}$$

depending on the stencil

$$\frac{\partial}{\partial \vec{a_s}} \sum_{j \in s_k} \|\vec{e_{s_k j}}\|^2 = \begin{cases} \displaystyle\sum_{j \in s_k} \frac{\partial \|\vec{e_{s_k j}}\|^2}{\partial \vec{a_s}} & \text{if } s = s_k \\ 0 & \text{if } s \neq s_k \end{cases} \tag{50}$$

equation (49) becomes

$$\frac{\partial E}{\partial \vec{a_s}} = \sum_{j \in s} \frac{\partial \|\vec{e_{sj}}\|^2}{\partial \vec{a_s}} \tag{51}$$

$$\sum_{j \in s} \frac{\partial \|\vec{e_{sj}}\|^2}{\partial \vec{a_s}} = \sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \left( \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \vec{v_j} + \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} \vec{a_s} + \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \vec{b_s} \right) = 0 \Rightarrow$$

$$\left( \sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} \right) \vec{a_s} + \left( \sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \right) \vec{b_s} = -\sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \vec{v_j} \tag{52}$$

Similarly for $\dfrac{\partial E}{\partial \vec{b_s}} = 0$ we end up with:

$$\left( \sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} \right) \vec{a_s} + \left( \sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \right) \vec{b_s} = -\sum_{j \in s} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \vec{v_j} \tag{53}$$

Combining equations (52) (53) into a more compact form:

$$\mathbb{M}_s \begin{bmatrix} \vec{a_s} \\ \vec{b_s} \end{bmatrix} = -\sum_{j \in s} \begin{bmatrix} \left[\dfrac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \dfrac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \\ \hline \left[\dfrac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right]^T \dfrac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \end{bmatrix} \vec{v_j} = -\sum_{j \in s} \mathbb{H}_{js}^T \vec{v_j} \tag{54}$$

Where

$$\mathbb{M}_s = \left[\begin{array}{c|c} \displaystyle\sum_{j\in s}\left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} & \displaystyle\sum_{j\in s}\left[\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \\ \hline \displaystyle\sum_{j\in s}\left[\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} & \displaystyle\sum_{j\in s}\left[\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} \end{array}\right] \tag{55}$$

is a symmetric 6x6 matrix.

### 5.1 eliminating the node velocities $\vec{v_j}$

From (48) we have:

$$\vec{v_j} = -\sum_{s\ni j} \mathbb{N}_j^{-1}\mathbb{H}_{js}\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} \tag{56}$$

and replacing $\vec{v_j}$ in equation (54) (the two relations do not necessarily concern the same stencil, that's why an $s'$ is introduced),

$$\begin{aligned}
\mathbb{M}_s\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} &= -\sum_{j\in s}\mathbb{H}_{js}^T\left(-\sum_{s'\ni j}\mathbb{N}_j^{-1}\mathbb{H}_{js'}\begin{bmatrix}\vec{a_{s'}}\\ \vec{b_{s'}}\end{bmatrix}\right) \Rightarrow \\
\mathbb{M}_s\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} &= -\sum_{j\in s}\left(-\sum_{s'\ni j}\mathbb{H}_{js}^T\mathbb{N}_j^{-1}\mathbb{H}_{js'}\begin{bmatrix}\vec{a_{s'}}\\ \vec{b_{s'}}\end{bmatrix}\right) \Rightarrow \\
\mathbb{M}_s\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} &= \sum_{s'}\left(\sum_{j\in s\cap s'}\mathbb{H}_{js}^T\mathbb{N}_j^{-1}\mathbb{H}_{js'}\begin{bmatrix}\vec{a_{s'}}\\ \vec{b_{s'}}\end{bmatrix}\right) \Rightarrow \\
\mathbb{M}_s\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} &- \sum_{s'}\left(\underbrace{\sum_{j\in s\cap s'}\mathbb{H}_{js}^T\mathbb{N}_j^{-1}\mathbb{H}_{js'}}_{a_{ss'}\ s\neq s'}\right)\begin{bmatrix}\vec{a_{s'}}\\ \vec{b_{s'}}\end{bmatrix} = 0
\end{aligned} \tag{57}$$

### 5.2 eliminating stencil velocities $\vec{a_s}, \vec{b_s}$

Starting from equation (54)

$$\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} = -\sum_{j\in s}\mathbb{M}_s^{-1}\mathbb{H}_{js}^T\ \vec{v_j} \tag{58}$$

Changing the node index to i (instead of j so far), since it's not the same nodes we are referring to, equation (48) becomes

$$\mathbb{N}_i\vec{v_i} = -\sum_{s\ni i}\mathbb{H}_{is}\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix} \tag{59}$$

Replacing $\begin{bmatrix}\vec{a_s}\\ \vec{b_s}\end{bmatrix}$ as it is expressed in (58) in the above equation yields

$$\mathbb{N}_i\vec{v_i} - \sum_j \left( \underbrace{\sum_{s\ni\{i,j\}} \mathbb{H}_{is}\mathbb{M}_s^{-1}\mathbb{H}_{js}^T}_{a_{ij}\ i\neq j} \right) \vec{v_j} = 0 \tag{60}$$

### 5.3 Elaborating on some expressions

So far we referred generically to some expressions without getting into the details of computing them or making them more explicit. Finally, this subsection is detailing this aspect. Let us first consider the partial derivatives of $\vec{e_{sj}}$ :

$$\frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} = \sqrt{w_s\mu_{sj}}\ \mathbb{I}_{(3\times 3)} \tag{61}$$

$$\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} = -\sqrt{w_s\mu_{sj}}\ \mathbb{I}_{(3\times 3)} \tag{62}$$

$$\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}} = \sqrt{w_s\mu_{sj}} \begin{bmatrix} 0 & c_{s3}-x_{j3} & x_{j2}-c_{s2} \\ x_{j3}-c_{s3} & 0 & c_{s1}-x_{j1} \\ c_{s2}-x_{j2} & x_{j1}-c_{s1} & 0 \end{bmatrix} \tag{63}$$

It is worth noting that $\frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}}$ , $\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}}$ are diagonal and $\frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}$ is antisymmetric. Using a more compact notation

$$\begin{aligned}\frac{\partial \vec{e_{sj}}}{\partial (\vec{a_s},\vec{b_s})} &= \begin{bmatrix}\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} & \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\end{bmatrix} \\ &= \sqrt{w_s\mu_{sj}} \begin{bmatrix} -1 & 0 & 0 & 0 & c_{s3}-x_{j3} & x_{j2}-c_{s2} \\ 0 & -1 & 0 & x_{j3}-c_{s3} & 0 & c_{s1}-x_{j1} \\ 0 & 0 & -1 & c_{s2}-x_{j2} & x_{j1}-c_{s1} & 0 \end{bmatrix}\end{aligned} \tag{64}$$

In light of the differentiation of $\vec{e_{sj}}$ the expressions of $\mathbb{N}_j$ and $\mathbb{H}_{js}$ now become clearer

$$\begin{aligned}&\mathbb{H}_{js} = \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}}\right]^T \begin{bmatrix}\frac{\partial \vec{e_{sj}}}{\partial \vec{a_s}} & \frac{\partial \vec{e_{sj}}}{\partial \vec{b_s}}\end{bmatrix} \Rightarrow \\ &\mathbb{H}_{js} = w_s\mu_{sj} \begin{bmatrix} -1 & 0 & 0 & 0 & c_{s3}-x_{j3} & x_{j2}-c_{s2} \\ 0 & -1 & 0 & x_{j3}-c_{s3} & 0 & c_{s1}-x_{j1} \\ 0 & 0 & -1 & c_{s2}-x_{j2} & x_{j1}-c_{s1} & 0 \end{bmatrix}\end{aligned} \tag{65}$$

$$\begin{aligned}\mathbb{N}_j = \sum_{s\ni j} \left[\frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}}\right]^T \frac{\partial \vec{e_{sj}}}{\partial \vec{v_j}} \Rightarrow \\ \mathbb{N}_j = \sum_{s\ni j} w_s\mu_{sj}\mathbb{I}_{(3\times 3)}\end{aligned} \tag{66}$$

We denote

$$\mathbb{M} = \mathrm{diag}(\mathbb{M}_s) \tag{67}$$
$$\mathbb{N} = \mathrm{diag}(\mathbb{N}_i) \tag{68}$$
$$\mathbb{H} = (\mathbb{H}_{js})_{1 \leq s \leq n_{stencil}}^{1 \leq j \leq n_{nodes}} \tag{69}$$

As an example, eliminating $u$ as shown in (39), would mean

$$\mathbb{M} = \mathbb{A}_{(a|b)(a|b)} \tag{70}$$
$$\mathbb{H} = \mathbb{A}_{u(a|b)} \tag{71}$$
$$\mathbb{N} = \mathbb{A}_{uu} \tag{72}$$

Thus, the coefficient matrices indexed according to the two different cases of elimination (38) are given:

$$\mathbb{A}_u = \mathbb{N} - \mathbb{H}\mathbb{M}^{-1}\mathbb{H}^T \tag{73}$$
$$\mathbb{A}_{(a|b)} = \mathbb{M} - \mathbb{H}^T\mathbb{N}^{-1}\mathbb{H} \tag{74}$$

## 6 CONCLUSIONS

- In the present article, a novel method, Rigid Motion Mesh Morpher has been introduced. We provided indications, according to which, it handles intrinsically, at the same time, the rotations and the mesh anisotropy of any mesh, since its design is mesh-less.
- It necessitates the solution of a sparse, symmetric positive definite (SPD) problem, proportional to the size of the mesh.
- Ongoing work: we need to validate the approach on a variety of test cases and compare it to existing approaches with the hope to demonstrate its superiority.
- It will be implemented as a part of an adjoint solver loop.
- Future work may include smoothing of the surface node velocity field as provided by the adjoint solver or equivalently a CAD-free parametrization of the surface.

## 7 ACKNOWLEDGEMENTS

This work was funded by the EU through the FP7-PEOPLE-2012-ITN "AboutFlow" Grant agreement number:317006.

## REFERENCES

[1] Carlo L. Botasso, David Detomi, and Roberto Serra. The Ball-Vertex Method: a New Simple Spring Analogy Method for Unstructured Dynamic Meshes. *Computer Methods in Applied Mechanics and Engineering*, 194:4244–4264, 2005.

[2] Christoph Degand and Charbel Farhat. A three-dimensional torsional spring analogy method for unstructured dynamic meshes. *Computers & structures*, 80(3):305–316, 2002.

[3] Richard P. Dwight. Robust Mesh Deformation using the Linear Elasticity Equations. In *Computational Fluid Dynamics 2006*, pages 401–406. Springer, 2009.

[4] Ch Farhat, C Degand, B Koobus, and M Lesoinne. Torsional springs for two-dimensional dynamic unstructured fluid meshes. *Computer methods in applied mechanics and engineering*, 163(1):231–245, 1998.

[5] David A Field. Laplacian smoothing and Delaunay triangulations. *Communications in applied numerical methods*, 4(6):709–712, 1988.

[6] Peter Hansbo. Generalized laplacian smoothing of unstructured grids. *Communications in numerical methods in engineering*, 11(5):455–464, 1995.

[7] Stefan Jakobsson and Olivier Amoignon. Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization. *Computers & Fluids*, 36(6):1119–1136, 2007.

[8] AM Morris, CB Allen, and TCS Rendall. Cfd-based optimization of aerofoils using radial basis functions for domain element parameterization and mesh deformation. *International journal for numerical methods in fluids*, 58(8):827–860, 2008.

[9] Nilanjan Mukherjee. A Hybrid, Variational 3D Smoother for Orphaned Shell Meshes. In *IMR*, pages 379–390. Citeseer, 2002.

[10] Marco Andrea Pischedda. Mesh Morphing techniques based on continuum deformation models. Master's thesis, Politecnico di Milano, 2008.

[11] Z. Su, S. Wang, C. Yu, F. Liu, and X. Shi. A Novel Laplacian Based Method for Mesh Deformation. *Journal of Information and Computational Science*, 7(4):877–883, 2010.

[12] Anh H Truong, Chad A Oldfield, and David W Zingg. Mesh movement for a discrete-adjoint Newton-Krylov algorithm for aerodynamic optimization. *AIAA journal*, 46(7):1695–1704, 2008.

[13] Shaoting Zhang, Junzhou Huang, and Dimitris N Metaxas. Robust mesh editing using Laplacian coordinates. *Graphical Models*, 73(1):10–19, 2011.