7. Appendix

7.1. Active Surface Model: Continuous Formulation

Our objective is to minimize the total energy $\mathbf{E}$ in Eq. 2. There is no analytical solution for the global minimum of $\mathbf{E}$. But, as mentioned in Section 3.1, any local minimum must satisfy the associated Euler-Lagrange equation given in Eq. 4. To find a surface that does this, surface evolution is used by introducing a time t parameter into 4 and writing

$$\frac{\partial v(s,r,t;\Phi)}{\partial t} + L(v(s,r,t;\Phi)) = F(v(s,r,t;\Phi)) \ , \quad (14)$$

where $L(v(s,r,t;\Phi))$ is the R.H.S of Eq. 4.

Solving 14, requires specifying an initial surface. Earlier approaches [9, 19] used a manual initialization, whereas in [27, 5] another model is used to predict the initial curve. To ensure the reached local minima corresponds to the desired curve, these approaches require the initialization to be close to the target shape. In DASM, we rely instead on the graph-convolution layers to provide a good initialization.

7.2. Active Surface Model: Discrete Formulation

In the continuous formulation of Section 3.1, computing the solution to Eq. 4 requires computing the derivatives of order 2 and 4 for the mapping v of Eq. 1. To compute them in practice, we discretize the surface and use finite difference equations to estimate the derivatives. Given a small value of δs, finite-difference approximations for the derivatives w.r.t s can be written as,

$$\begin{aligned}
\frac{\partial v}{\partial s} &\approx \frac{1}{\delta s}[v(s+\delta s,r) - v(s,r)] \ , \\
\frac{\partial^2 v}{\partial s^2} &\approx \frac{1}{\delta s^2}[v(s+\delta s,r) - 2v(s,r) + v(s-\delta s,r)] \ , \\
\frac{\partial^3 v}{\partial s^3} &\approx \frac{1}{\delta s^3}[v(s+2\delta s,r) - 3v(s+\delta s,r) \\
&\qquad + 3v(s,r) - v(s-\delta s,r)] \ , \\
\frac{\partial^4 v}{\partial s^4} &\approx \frac{1}{\delta s^4}[v(s+2\delta s,r) - 4v(s+\delta s,r) \\
&\qquad + 6v(s,r) - 4v(s-\delta s,r) + v(s-2\delta s,r)] \ ,
\end{aligned}$$

Similarly, we can write finite difference equations w.r.t r as well.

Now to compute these approximations, we need to compute $v(s+\delta s)$ and other similar terms. Let us therefore take (s,r) be the 2D coordinates that v maps to the coordinates of a specific vertex. In an irregular grid, $(s, r+\delta r)$, $(s+\delta s, r)$, or any of s, r coordinates that appear in the derivative computations will in general *not* be be mapped to another vertex for any choice of $\delta s, \delta r$. Fig. 8 illustrates their actual positions depending on the number of neighbors the vertex has.

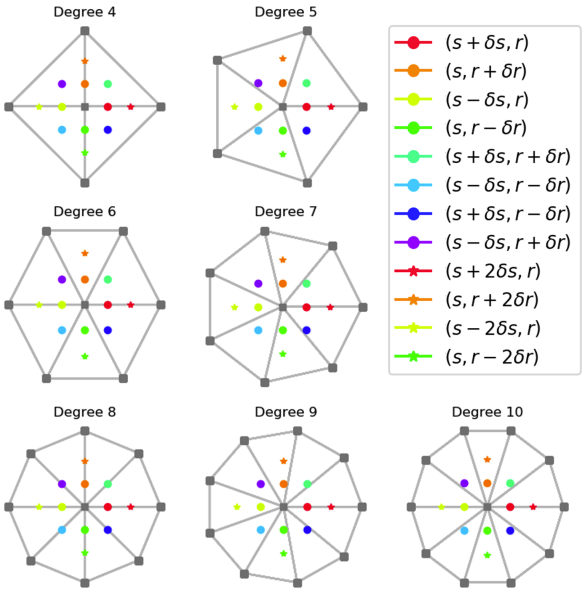

Figure 8: **Finite Differences.** Relative positions of $(s + k_1\delta, r + k_2\delta)$ terms w.r.t (s,r) which is at the center and its 1-ring neighbors from degree 4 to 10.

We can nevertheless compute the 3D coordinates they map to as follows. Let us first consider the 3D point $v(s+\delta s, r)$ that $(s+\delta s, r)$ gets mapped to and it is depicted by orange circle in Fig. 2. For δs small enough, it belongs to a facet of which $v(s,r)$ is a vertex and let $v(s_1,r_1)$ and $v(s_2,r_2)$ be the other two. We can compute the barycentric coordinates λ, λ_1, and λ_2 of $v(s+\delta s, r)$ in that facet by solving

$$\begin{bmatrix} s+\delta s \\ r \\ 1 \end{bmatrix} = \begin{bmatrix} s & s_1 & s_2 \\ r & r_1 & r_2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda \\ \lambda_1 \\ \lambda_2 \end{bmatrix} \ . \quad (15)$$

Given these barycentric coordinates, we can now estimate $v(s+\delta s, r)$ as

$$\lambda * v(s,r) + \lambda_1 * v(s_1,r_1) + \lambda_2 * v(s_2,r_2) \ , \quad (16)$$

which allows us to estimate $\frac{\partial v}{\partial s}$ according to the above finite-difference equations. For this approximation to be valid, we pick δs such that all terms in finite-difference expressions lie within the 1-ring neighborhood of $v(s,r)$. We can repeat the process for all the other expressions involving δs in these equations and, hence, compute all required derivatives. Regular square and hexagonal grids are special cases in which these computations can be simplified.

7.3. Matrix Inversion using Neumann Series

We are approximating the inverse of $(\mathbf{A} + \alpha\mathbf{I})^{-1}$ using the Neumann series given in Eq. 7. In Fig 9, we plot both

RMSE in estimating the inverse and the time it takes to perform the estimation as a function of K. Given the trade off between running time and accuracy, we pick $K = 4$ for the estimation.

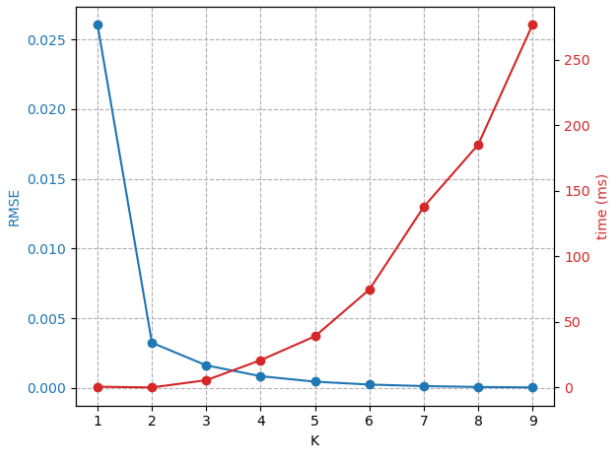

Figure 9: **Neumann Series Approximation.** RMSE between the approximated inverse and the true one in blue and computation time in red as function of K. $K = 4$ gives an acceptable trade-off between the two.

7.4. Quantitatively Measuring Mesh Regularization with Consistency Metrics

All the metrics used in Sec. 4 evaluate the accuracy of the meshes. We use them because they are the standard metrics used in the literature. But to get a better understanding of the quality of the meshes, we provide two more metrics; mean edge length and mean surface Laplacian. We observe that around abnormalities such as those highlighted by orange arrows in Fig. 5, 7, the edge lengths and surface Laplacians tend to increase significantly. This increases mean edge length and mean surface Laplacian and its effect can be seen in Table 7.

$\lambda_{e.}$		Chf. (↓)	Edg. Length	Surf. Lap.
1.0	Mesh R-CNN	0.232	0.023 ± 0.011	0.033 ± 0.031
	Ad.-DASM	**0.231**	0.022 ± 0.009	0.023 ± 0.019
0.6	Mesh R-CNN	0.212	0.024 ± 0.015	0.045 ± 0.045
	Ad.-DASM	**0.206**	0.023 ± 0.011	0.029 ± 0.020
0.2	Mesh R-CNN	0.189	0.028 ± 0.021	0.066 ± 0.075
	Ad.-DASM	**0.183**	0.025 ± 0.015	0.037 ± 0.049
0.0	Mesh R-CNN	**0.144**	0.141 ± 0.142	0.708 ± 0.671
	Ad.-DASM	0.167	0.070 ± 0.072	0.254 ± 0.276

Table 7: **Results on ShapeNet** as a function of λ_{edge}. Note that this is an extension of Table 2.