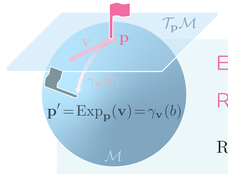
Parallelised Differentiable Straightest Geodesics for 3D Meshes

Hippolyte Verninas Caner Korkmaz Stefanos Zafeiriou Tolga Birdal* Simone Foti*

IMPERIAL

CVPR JUNE 3-7, 2026 DENVER COLORADO



Euclidean Space ($\mathbb{R}^3$): translation via vector addition $\mathbf{p} + \mathbf{v}$.

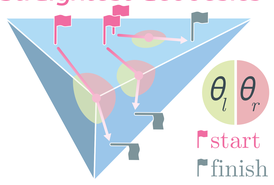
Riemannian Surfaces ($\mathcal{M} \subset \mathbb{R}^3$): translation via $\mathrm{Exp}_{\mathbf{p}}(\mathbf{v}) = \gamma_{\mathbf{v}}(b)$.

Requires solving IVP ODE for geodesics $\gamma_{\mathbf{v}}(t) = \begin{cases} \nabla_{\dot\gamma(t)}\dot\gamma(t) = 0 \\ \gamma(a) = \mathbf{p} \\ \dot\gamma(a) = \mathbf{v} \end{cases}$

Meshes ($\mathcal{M} \rightsquigarrow \mathcal{M}$): translation still via $\mathrm{Exp}_{\mathbf{p}}(\mathbf{v})$, but surface is discrete, not analytically defined, and ODE solvers not suitable.

How do we compute the Exp map on meshes

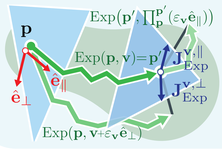
Straightest Geodesics



$\nabla_{\dot\gamma(t)}\dot\gamma(t) = 0 \Rightarrow \theta_l = \theta_r \quad (\forall \mathbf{p} \in \mathscr{M})$

DIFFERENTIABLE STRAIGHTEST GEODESICS

2 methods for differentiating through the Exp map to enable the integration of discrete Riemannian geometry in modern learning and optimisation pipelines for meshes.



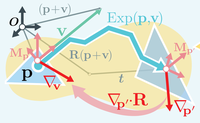
Geodesic Finite Differences (GFD) for $\mathbf{v}$

$$\mathbf{J}^{\mathbf{v}}_{\mathrm{Exp}} \approx \mathbf{M}^{\dagger}_{\mathbf{p}'} \cdot \left[\frac{\mathrm{Exp}(\mathbf{p}, \mathbf{v}+\varepsilon_{\mathbf{v}}\hat{\mathbf{e}}_\parallel) - \mathbf{p}'}{\varepsilon_{\mathbf{v}}} \quad \frac{\mathrm{Exp}(\mathbf{p}, \mathbf{v}+\varepsilon_{\mathbf{v}}\hat{\mathbf{e}}_\perp) - \mathbf{p}'}{\varepsilon_{\mathbf{v}}} \right]$$

$$\mathbf{M}^{\dagger}_{\mathbf{p}'} = (\mathbf{M}_{\mathbf{p}}^{\top}\mathbf{M}_{\mathbf{p}'})^{-1}\mathbf{M}_{\mathbf{p}}^{\top} \qquad \mathrm{Exp}(\mathbf{p}, \mathbf{v}+\varepsilon_{\mathbf{v}}\hat{\mathbf{e}}_\parallel) = \mathrm{Exp}(\mathbf{p}', \prod_{\mathbf{p}}^{\mathbf{p}'}(\varepsilon_{\mathbf{v}}\hat{\mathbf{e}}_\parallel))$$

$$\mathbf{M}_{\mathbf{p}} = [\hat{\mathbf{e}}_u \; \hat{\mathbf{e}}_v]$$

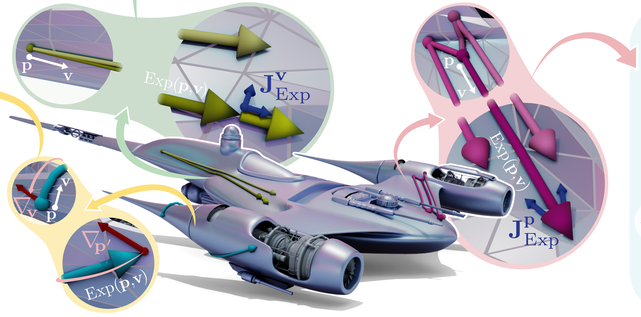
Extrinsic Proxy (EP)



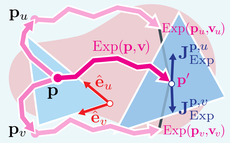
$$\mathbf{R} = \mathbf{M}_{\mathbf{p}}^{\top}\mathbf{M}_{\mathbf{p}'} \qquad t = \mathbf{p}' - \mathbf{R}\cdot(\mathbf{p}+\mathbf{v})$$

$$\mathbf{p}'_{\mathrm{proxy}} = \varphi(\mathbf{p}, \mathbf{v}) = [\mathbf{R}^{\mathrm{fix}}\cdot(\mathbf{p}+\mathbf{v})] + \mathbf{t}^{\mathrm{fix}}$$

$$\mathbf{J}^{\mathbf{v}}_{\mathrm{Exp}} = \mathbf{R}^{\mathrm{fix}}\cdot\mathbf{I} \qquad \mathbf{J}^{\mathbf{p}}_{\mathrm{Exp}} \to \mathbf{0}$$



GFD for $\mathbf{p}$



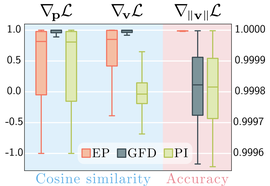
$$\mathbf{J}^{\mathbf{p}}_{\mathrm{Exp}} \approx \mathbf{M}^{\dagger}_{\mathbf{p}'} \cdot \left[\frac{\mathrm{Exp}(\mathbf{p}_u,\mathbf{v}_u) - \mathbf{p}'}{\varepsilon_{\mathbf{p}}} \quad \frac{\mathrm{Exp}(\mathbf{p}_v,\mathbf{v}_v) - \mathbf{p}'}{\varepsilon_{\mathbf{p}}} \right]$$

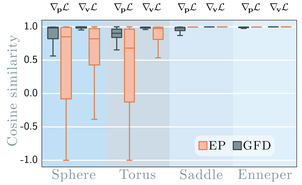
$$\mathbf{p}_u = \mathrm{Exp}(\mathbf{p}, \varepsilon_{\mathbf{p}}\hat{\mathbf{e}}_u) \qquad \mathbf{p}_v = \mathrm{Exp}(\mathbf{p}, \varepsilon_{\mathbf{p}}\hat{\mathbf{e}}_v)$$

$$\mathbf{v}_u = \prod_{\mathbf{p}}^{\mathbf{p}_u}(\mathbf{v}) \qquad \mathbf{v}_v = \prod_{\mathbf{p}}^{\mathbf{p}_v}(\mathbf{v})$$

ACCURATE GRADIENTS

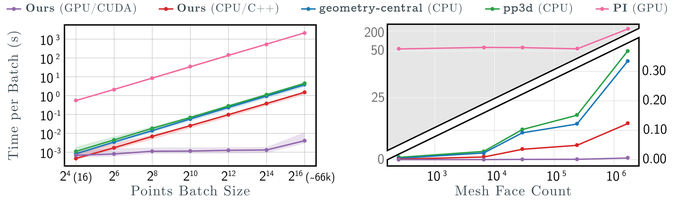
compared to closed form Exp on the sphere and solving ODEs





GPU PARALLELISATION

for computation on large batches and meshes

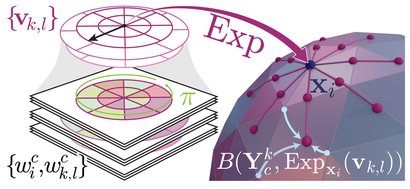


NOVEL APPLICATIONS

AGC

Adaptive Geodesic Convolutions

Patches dynamically learn their receptive field per layer and channel during training.



Filter weights on polar chart defined as:
$\mathbf{v}_{k,l} = k\rho_w \mathbf{R}(\theta_l)\hat{\mathbf{v}}_{base}$ for $k = 1...n_\rho,\ l = 1...n_\theta$

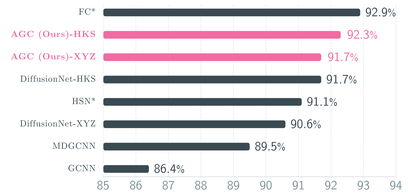
Exp map and Barycentric interpolation to convolve:

$$\mathbf{Y}^{\kappa+1}_{i,c} = \mathbf{Y}^{\kappa}_{i,c} w^c_i + \max_{\pi} \sum_{l=1}^{n_\theta}\sum_{k=1}^{n_\rho} w^c_{\pi(l),k} B(\mathbf{Y}^{\kappa}_c, \mathrm{Exp}_{\mathbf{x}_i}(\mathbf{v}_{k,l}))$$

rotational invarince

Filters are grouped in subsets sharing same $\{\mathbf{v}_{k,l}\}$. ρ_w are learned differentiating with EP.

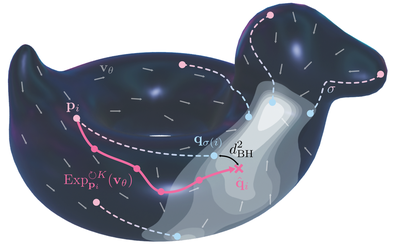
BODY SEGMENTATION RESULTS



MeshFlow

A Better Flow Matching on Meshes

Learn a static vector field to geodesically move points in K-steps of Exp map.



During training, we minimise the biharmonic distance to precomputed OT couplings.

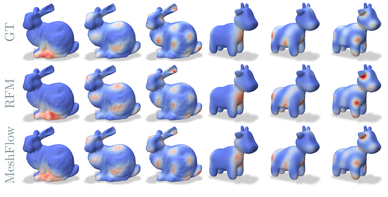
$$\mathcal{L}(\theta) = \mathbb{E}_{\substack{\mathbf{p}_i\sim\mathcal{U}(\mathscr{M}) \\ \mathbf{q}_i\sim\mathcal{Q}(\mathscr{M})}} \left[\sum_{i=1}^{B} d^2_{\mathrm{BH}}\left(\mathrm{Exp}^{\circ K}_{\mathbf{p}_i}(\mathbf{v}_\theta), \mathbf{q}_{\sigma(i)}\right) \right]$$

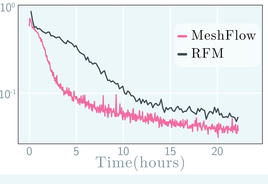
$$\sigma = \underset{\hat\sigma\in\mathcal{S}}{\mathrm{argmin}} \sum_{i=1}^{B_c} d^2_{BH}(\mathbf{p}_i, \mathbf{q}_{\hat\sigma(i)}) \leftarrow \text{OT coupling}$$

DISTRIBUTION FITTING RESULTS

vs RFM

16,000× faster inference	97% less GPU memory
46.6% lower KL div.	8.4% lower BH Cham dist.





Mesh-LBFGS

A 2nd Order Optimiser on Meshes

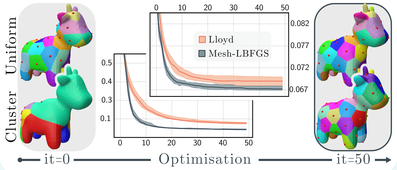
This Quasi-Newton method requires multiple Exp and parallel transport evaluations per iteration. This is computationally tractable only thanks to our parallel GPU implementation.

We apply it to the Geodesic Centroidal Voronoi Tesselation

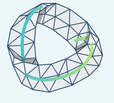
TESSELATION RESULTS

Compared to standard Lloyd's algorithm:

>2× faster convergence
16% lower final energy
recovers from por initialisazions



pip install digeo



digeo