STAG: Smart Tools and Applications in Graphics (2024)
T. G. Dulecha, U. Castellani, and A. Giachetti (Editors)

Mesh Comparison Using Regular Grids

Patrizia Kaye and Ioannis Ivrissimtzis[1]

[1]University of Durham, UK

Abstract

A symmetric grid-based approach to mesh comparison is proposed, providing intuitive visual results alongside an objective measure of the local differences between meshes. The difference function is defined on the nodes of a regular 3D lattice, making it suitable as input for a variety of analysis algorithms. The visual results are compared and comparable to the Metro tool.

CCS Concepts

• ***Computing methodologies*** → ***Mesh geometry models;*** *Shape analysis; Mesh models;*

1. Introduction

When comparing two meshes, the discussion is often based on visual inspection or on numerical computations[ZP01; RFT04; SPA*14; SGSH23]. Here, a simple, fast grid-based algorithm is proposed to compute a symmetric local distance function between two meshes A and B. The domain of this function is not a set of attributes of one mesh, but a regular grid which provides a common reference frame. The proposed algorithm detects and visualises local differences between two irregular meshes in a structured way with the potential to facilitate to deeper analysis. Moreover, the algorithm is fast, as the regular lattice structure allows the efficient computation of nearest neighbours to its nodes.

2. Mesh Comparison Algorithm

The aim is to compare a mesh A with a processed form thereof, B. The local difference function is defined on a $N_x \times N_y \times N_z$ lattice $\mathbf{L}$ (regular grid) with integer coordinates. The value of N_x is chosen by the user, and determines the resolution at which the comparison will be performed. Mesh A is scaled such that the difference between the minimum and the maximum x-coordinate is N_x, then N_y and N_z are defined by A's maximum coordinate differences (plus a small buffer) along the y and z directions. Each node, $n \in \mathbf{L}$ stores its shortest distance to each mesh: d_A^n and d_B^n. As an optimisation, d_A^n and d_B^n are only assigned for nodes within a certain distance from the meshes.

We iterate through all faces of the input mesh, say, mesh A, and for each face f, compute the smallest parallelepiped bounding box P aligned to the axes of $\mathbf{L}$. P is then grown by one node in each direction to allow for an overall smoother minimum distance function to defined on the nodes of $\mathbf{L}$. Each $n \in P$ has its distance to f calculated by using the method in [MWJ95], and d_A^n is updated to the smaller of its current value and the new distance.

After this is done for all $n \in P$, we proceed to the next face in A. Once completed for both A and B, the lattice is fully trained and can be used directly, or exported for later analysis.

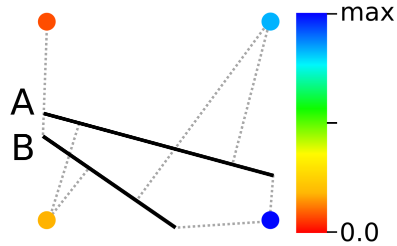

Figure 1: *Comparison of distances between nodes and two meshes. The difference in the dotted line lengths is reflected by the hue of the node, which is then transferred to the mesh vertices. Red corresponds to no difference, and blue to the largest value obtained.*

2.1. Visualisation of Local Differences

To visualise the differences between A and B, hues are applied to areas of significance. For each mesh vertex $v \in A$, the nearest grid node n is found and the difference in distances $|d_A^n - d_B^n|$ mapped to a hue according to its magnitude. We chose to assign hues on a per-mesh relative scale, displayed in Figure 1, with red indicating no difference and dark blue indicating the largest difference between meshes, following Metro's default colour scheme. Vertices closest to a lattice node trained by one only of the input meshes are assigned the hue indicating maximum difference. The local difference function is thus a symmetric function defined on the nodes of $\mathbf{L}$, but mapped and visualised in two different ways, one on mesh A and one on B. Two coloured meshes are therefore produced: A with the differences to B highlighted, and B with the differences to A highlighted.

DOI: 10.2312/stag.20241347

2.2. Mean Absolute Difference

We propose the Mean Absolute Difference (MAD) over all trained nodes as a global measure of differences, which can be computed from the local difference function defined on **L**

$$MAD = \frac{1}{N}\sum_{i}^{N} |d_A^i - d_B^i| \quad (1)$$

Nodes trained by only one input are assigned a nominal distance of $2\sqrt{3}$ – the minimum distance from a mesh at which we are guaranteed to record a distance on that node. Values $> 2\sqrt{3}$ may nevertheless be recorded, due to how P is computed, depending on the size and orientation of the face contained therein. The measure is symmetric and intuitive – the larger the differences at each node, the larger the global difference will be. We use the term *mean absolute difference* instead of the more common *mean absolute error* to avoid the implication that one mesh is the "ground truth", when this may not be the case in practice.

3. Visual Results

The visual comparison results are shown in Figure 2 and are very similar to those achieved with Metro[CRS98]: differences are typically in the smoothness of the rendering.

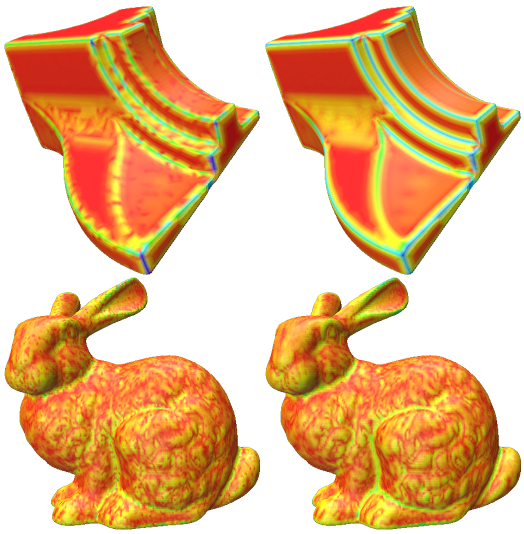

Figure 2: *Differences from smoothing the Fandisk and Bunny, highlighted by the proposed algorithm (left) and Metro (right)[Sha*19]. The maximum difference for the Fandisk and Bunny was was 3.4295 and 1.5683 respectively.*

4. Numerical Validation

To validate the MAD between meshes, noise was added to four standard benchmark meshes (the Fandisk, Bunny, Armadillo, and Happy Buddha) by random vertex displacement[CCC*08; MC21] in increments of 0.1% up to 1.0% In line with intuition, Figure 4 shows that as the level of noise increases, so do the number of green highlighted areas (indicating proximity to a partially-trained node), relatively uniformly across the surface of the mesh. The symmetric Hausdorff distance[GBK05] increases almost linearly with noise.

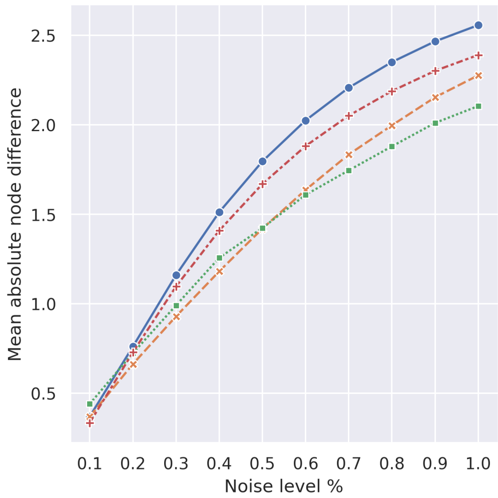

Figure 3: *MAD difference increasing with noise level[Was21; McK10]. The lines are for the Happy Buddha (blue), Armadillo (orange), Fandisk (green), and Bunny (red) models.*

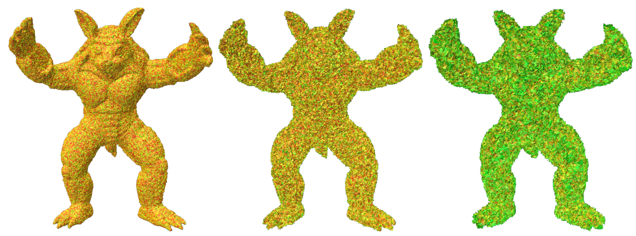

Figure 4: *0.1%, 0.5%, and 1.0% noise applied to the Armadillo[Sha*19]. The maximum differences were 0.5857, 3.9026, and 4.5568 respectively (symmetric Hausdorff: 0.6483, 3.1655 and 6.3412).*

5. Contributions

A simple, fast, symmetric comparison algorithm that computes a numerical difference between two meshes and provides a visual representation thereof. The numerical difference is computed with respect to a common reference lattice and a preliminary investigation is presented on how to exploit the highly-structured and exportable data recorded on the lattice's nodes.

6. Future Work

We plan to investigate methods for determining the best resolution at which to run the analysis – even if the optimal resolution cannot be proven, a heuristic measure could nevertheless be useful. Storing vector (rather than scalar) distances from an input to the lattice would also provide additional information for analysis.

References

[CCC*08] CIGNONI, PAOLO, CALLIERI, MARCO, CORSINI, MASSIMILIANO, et al. "MeshLab: an Open-Source Mesh Processing Tool". *Eurographics Italian Chapter Conference*. Ed. by SCARANO, VITTORIO, CHIARA, ROSARIO DE, and ERRA, UGO. The Eurographics Association, 2008. ISBN: 978-3-905673-68-5. DOI: 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136 2.

[CRS98] CIGNONI, PAOLO, ROCCHINI, CLAUDIO, and SCOPIGNO, ROBERTO. "Metro: Measuring Error on Simplified Surfaces". *Comput. Graph. Forum* 17.2 (1998), 167–174 2.

[GBK05] GUTHE, MICHAEL, BORODIN, PAVEL, and KLEIN, REINHARD. "Fast and Accurate Hausdorff Distance Calculation between Meshes". *J. of WSCG* 13 (2005), 41–48 2.

[MC21] MUNTONI, ALESSANDRO and CIGNONI, PAOLO. *PyMeshLab*. Jan. 2021. DOI: 10.5281/zenodo.4438750 2.

[McK10] MCKINNEY, WES. "Data Structures for Statistical Computing in Python". *Proceedings of the 9th Python in Science Conference*. Ed. by van der WALT, STÉFAN and MILLMAN, JARROD. 2010, 56–61. DOI: 10.25080/Majora-92bf1922-00a 2.

[MWJ95] M.W.JONES. *3D Distance from a Point to a Triangle*. Tech. rep. CSR-5-95. Swansea University, Feb. 1995 1.

[RFT04] ROY, MICHAËL, FOUFOU, SEBTI, and TRUCHETET, FRÉDÉRIC. "Mesh comparison using attribute deviation metric". *Journal of Image and Graphics* 4 (2004), 1–14. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.99.2495 1.

[SGSH23] STOREIDE, MARKUS SEBASTIAN BAKKEN, GEORGE, SONY, SOLE, ADITYA SUNEEL, and HARDEBERG, JON YNGVE. "Statistical Evaluation of 3D Manifolds Shape Retention during Simplification Stages". *Archiving Conference* 20.1 (June 2023), 18–22. ISSN: 2161-8798. DOI: 10.2352/issn.2168-3204.2023.20.1.4 1.

[Sha*19] SHARP, NICHOLAS et al. *Polyscope*. www.polyscope.run. 2019 2.

[SPA*14] SCHMIDT, J., PREINER, R., AUZINGER, T., et al. "YMCA – Your mesh comparison application". *Visual Analytics Science and Technology (VAST), 2014 IEEE Conference on*. Oct. 2014, 153–162. DOI: 10.1109/VAST.2014.7042491 1.

[Was21] WASKOM, MICHAEL L. "seaborn: statistical data visualization". *Journal of Open Source Software* 6.60 (2021), 3021. DOI: 10.21105/joss.03021. URL: https://doi.org/10.21105/joss.03021 2.

[ZP01] ZHOU, LAIXIN and PANG, ALEX. "Metrics and visualization tools for surface mesh comparison". *Visual Data Exploration and Analysis VIII*. Vol. 4302. 2001, 99–110. DOI: 10.1117/12.424920. URL: http://dx.doi.org/10.1117/12.424920 1.