Discontinuity-Aware 2D Neural Fields: Supplemental document

YASH BELHE, University of California San Diego, USA
MICHAËL GHARBI, Adobe Research, USA
MATTHEW FISHER, Adobe Research, USA
ILIYAN GEORGIEV, Adobe Research, UK
RAVI RAMAMOORTHI, University of California San Diego, USA
TZU-MAO LI, University of California San Diego, USA

In this supplemental document we provide insight into why grid-based methods are unsuitable to construct feature fields without approximations, as well as further implementation details of our method.

1 GRID REPRESENTATIONS SIMPLIFY CURVE TOPOLOGY

A potential approach to construct a feature field is to use a hybrid data structure containing both a grid and the set of discontinuity curves. This approach is commonly applied by classical feature-based texture methods [Ramanarayanan et al. 2004; Sen 2004; Tumblin and Choudhury 2004; Tarini and Cignoni 2005; Parilov and Zorin 2008] for representing sharp discontinuities in textures and images. These methods construct a regularly spaced grid and store the features at the corners of every grid cell. Reconstruction within each cell follows by interpolating features at the corners while avoiding smoothing across the discontinuities.

Once these methods pick a grid resolution, the locations of the grid vertices are fixed, and they cannot adapt to the topology of the curve network. This makes the approximation accuracy resolution dependent. More precisely, the four values at the corners can resolve at most four regions within each grid cell. At the cost of increasing resolution, the fraction of grid cells with more than four regions can be reduced. Nontheless, to correctly resolve the color in all regions, they require infinite subdivision, because any finite grid size can always contain cells with more than four regions, as shown in Fig. 1.

Therefore, existing methods usually have to modify the topology of the curve network, in a way that violates our continuity criteria (2) and (3) in paper Section 4.1. For example, the method of Parilov and Zorin [2008] requires each grid cell to contain a maximum of two discontinuity curves, and simplifies the curves to satisfy the constraints. Since we use a curved triangulation that adapts to the curve topology, we can achieve resolution independence without modifying the curve topology.

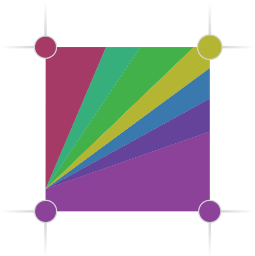

Fig. 1. A case that is difficult to handle using classical feature-based textures [Ramanarayanan et al. 2004; Sen 2004; Tumblin and Choudhury 2004; Tarini and Cignoni 2005; Parilov and Zorin 2008]. These methods store features at the corners of a grid cell, and reconstruct the continuous signal using edge-aware interpolation. However, it is impossible for these methods to handle certain topology when the discontinuity edges divide the space inside a grid into more than four regions (no matter how much we subdivide the grid), while preserving the desired smoothness and discontinuities.

2 IMPLEMENTATION DETAILS

Curved triangulation. TriWild's [Hu et al. 2019] curved triangulation allows us to specify parameters to control the mesh generation. The relative target edge length determines the resolution of the generated mesh. We use values between 0.001 and 0.1 depending on the signal complexity. The relative feature envelope determines the scale up to which the input discontinuity curves should be respected; we use values between 0.0001 for rendering and 0.001 for other applications, to ensure that most of the discontinuities are preserved by the triangulation (98.5% for the roses scene). Furthermore, we compress the linear triangle meshes for the rendering application using Draco (mesh connectivity is maintained and 0.0001% average error for the vertex positions in the hairball in Figure (1)); we have not experienced any performance degradation by doing so.

Feature assignment. We have three types of features in our feature field (Section 4.3): (1) isotropic features $\mathbf{F}$ associated with vertices, (2) clockwise/counterclockwise features $\mathbf{F}_{ij}^{\text{cw}}/\mathbf{F}_{ij}^{\text{ccw}}$ associated with directed edges, and (3) curve features $\mathbf{F}_{i,T}^{\text{curve}}$ associated with vertex-triangle pairs. All features are stored in a single array and each has five trainable parameters. The data structures defined below store indices into this array for each feature type. Most features are isotropic; for these, an array maps vertex indices to parameter indices. For the clockwise features we use a sparse matrix, where the i, j^{th} entry stores the index for the directed edge from i to j; we do the same for the counterclockwise feature. For the curve feature too, we use a sparse matrix, this time the i, j^{th} entry stores the index for the i^{th} vertex and j^{th} triangle.

Rasterization pipeline. While the data structures defined above are useful to logically connect features with trainable parameters, lookups into sparse matrices can lead to slow inference. For faster inference, we setup a few buffers per triangle that contain all the information necessary for feature interpolation within the triangle. For each discontinuous vertex in a triangle, we store a pointer to the parameter and the vertex position for the closest clockwise and counterclockwise feature. For each continuous vertex in a triangle, we store a pointer to the parameter for the corresponding feature. We also populate buffers with information about the Bézier curve control points if the triangle contains one. This pipeline allows more synchronous data fetching. We emulate this in PyTorch and expect much faster inference when done in a true rasterization pipeline.

REFERENCES

Yixin Hu, Teseo Schneider, Xifeng Gao, Qingnan Zhou, Alec Jacobson, Denis Zorin, and Daniele Panozzo. 2019. TriWild: Robust Triangulation with Curve Constraints. *ACM Trans. Graph. (Proc. SIGGRAPH)* 38, 4, Article 52 (2019), 15 pages.

Evgueni Parilov and Denis Zorin. 2008. Real-time rendering of textures with feature curves. *ACM Trans. Graph.* 27, 1 (2008), 1–15.

Ganesh Ramanarayanan, Kavita Bala, and Bruce Walter. 2004. Feature-Based Textures. In *Eurographics Workshop on Rendering*.

Pradeep Sen. 2004. Silhouette maps for improved texture magnification. In *Graphics Hardware*. 65–73.

Marco Tarini and Paolo Cignoni. 2005. Pinchmaps: textures with customizable discontinuities. *Comput. Graph. Forum (Proc. Eurographics)* (2005).

Jack Tumblin and Prasun Choudhury. 2004. Bixels: Picture Samples with Sharp Embedded Boundaries. *Rendering Techniques (Proc. EGWR)* (2004).