A Content-Aware Non-Uniform Grid for Fast Map Deformation

Pio Claudio
KAIST
pioclaudio@gmail.com

Inkyu An
KAIST
inkyu_ahn@kaist.ac.kr

Sung-eui Yoon
KAIST
sungeui@kaist.edu

Abstract

Existing map deformation techniques are mainly based on uniform grids and can be insufficient for real-time demands. For higher flexibility of allocating resolutions, we propose to use a content-aware non-uniform grid. Our content-aware non-uniform grid leads to adaptive resolutions on the map deformation. This adaptive approach results in both higher road deformability and performance. We implement our optimization in CUDA and test our approach in three different deformation map applications to demonstrate the effectiveness of our method.

1 Introduction

As various map data have become available and portable screens get ubiquitous, displaying digestible map information has also become a challenge, especially in applications generating route and tourist maps. Map users can have various tasks at hand at a given time, and maps can be tailored to specific tasks. Deformation has been a common approach used by recent map makers in creating different views for focus regions, forming shape patterns, cartograms, etc.

For creating a proper deformation, a user supplies the information and depending on that, a resulting personalized map is generated to address the current user need. It is, therefore, essential to formulate fast map transformations that enable interaction in switching views with transitions [1].

As more roads are included in a map, the time to process the map increases too. By deforming a uniform grid mesh of the map space instead of individual roads, the processing time can be lessened. However, choosing the resolution of the grid affects the quality of the deformation. In particular, optimization of a coarser mesh converges faster, but roads inside a quad may not be deformed well. On the other hand, optimization of a finer mesh deforms most roads well, but converges slower. In current approaches, this resolution parameter tuning is a trial-and-error process that could require repeated attempts.

Contributions. In this paper, we extend the uniform grid method and propose an adaptive approach, whose subdivision is determined by a significance function. We generate our non-uniform grid as a balanced grid, where each quad's neighbors have a single level size difference. We implement our optimization method in CUDA, and demonstrate how our approach can be easily integrated to existing approaches and improve their performance. We apply our approach to variable scaling in creating focus + context regions, transforming mental maps, and destination maps. In these applications, our method using adaptive grids shows more robust scaling performance than those using uniform grids, showing slower relative growth rates in computational time as a function of residue.

2 Background

In this section, we give a background to our work. Detailed reviews on prior approaches are available in the supplementary report.

In a simple road edge deformation, a road edge E undergoes a transformation $\mathbf{T}$, resulting to a deformed edge $\mathbf{T}E$. The goal of a basic edge deformation is to find a configuration of target edges such that the residual between the target, $\tilde{E}$, and the deformed edge $\mathbf{T}E$ are minimized. Specifically, the residual, D_{RE}, is defined as the

following:

$$D_{RE} = \sum_{E \in \mathbf{E}} \left\| \tilde{E} - \mathbf{T}E \right\|^2, \quad (1)$$

where $\mathbf{E}$ is a set containing all those edges. In this simple edge-based deformation, its computational time increases, as the number of road edges increases.

In order to accelerate this edge-based deformation process, a grid overlaid on a map is typically adopted in a map deformation for many different applications, because optimizing with the grid can provide higher performance than directly optimizing individual elements of maps. A grid overlaid on a map consists of quads that encompass the map elements (e.g., road vertices, road edges, Points-of-Interest (POIs), textures) under them.

Optimizing the grid indicates that we perform the process in terms of the grid components instead of individual road edges. In particular, grid components include quads Q, quad edges A, and quad vertices V. We assume that indices of two vertices of each road edge in Eq. 1 are E^1 and E^2, and those two vertices are then represented as barycentric coordinates associated with their respective quartet of quad corner vertices. The prior residual equation (Eq. 1) is then reformulated as the following:

$$D_{RE} = \sum_{E \in \mathbf{E}} \left\| \left(\sum_{p=1}^{4} b_p^{\tilde{E}^1} \tilde{V}_p^{\tilde{E}^1} - \sum_{p=1}^{4} b_p^{\tilde{E}^2} \tilde{V}_p^{\tilde{E}^2} \right) - \mathbf{T} \left(\sum_{p=1}^{4} b_p^{E^1} V_p^{E^1} - \sum_{p=1}^{4} b_p^{E^2} V_p^{E^2} \right) \right\|^2, \quad (2)$$

where $\tilde{V}$ is a target vertex and $b_p^{E^i}$ is a barycentric coordinate of a corner $V_p^{E^i}$ of a quad containing the edge endpoint indexed by E^i.

Deformation can deviate from the original shape of the map, so preservation of the shape of quads is applied [2]. Its objective, D_{QE}, called quad error, is defined as:

$$D_{QE} = \sum_{Q \in \mathbf{Q}} \sum_{i=1}^{4} \left\| \tilde{B} - \mathbf{K}\tilde{A}_i \right\|^2, \quad (3)$$

where $\tilde{A}_i$ is a deformed quad Q edge under $\mathbf{T}$, and $\tilde{B}$ is one of edges of the quad Q; the top edge is typically selected for this purpose. $\mathbf{K}$ is the original relationship computed by original edges, i.e., B and A_i.

Additionally, a smoothing objective can be considered to maintain the rigidity of the grid and implicitly avoid quad overlaps. The smoothing objective, D_{SM}, is a matrix form of the Laplacian smoothing [3]:

$$D_{SM} = \sum_{V \in \mathbf{V}} \left\| \mathbf{L}\tilde{V} \right\|^2, \quad (4)$$

$$\mathbf{L}_{ij} = \begin{cases} -1/Valence(V_i) & \text{if } V_i \text{ and } V_j \text{ are adjacent,} \\ 1 & \text{if } i = j, \\ 0 & \text{otherwise.} \end{cases}$$

Finally, the optimization of the map is achieved by minimizing the weighted sum of all the objectives:

$$D = w_{RE}D_{RE} + w_{QE}D_{QE} + w_{SM}D_{SM}. \quad (5)$$

Depending a type of applications, this overall objective can be appended with additional objectives (Sec. 4).

3 Our Method

Uniform grids are popular thanks to ease of use and fast results in deformation with a coarse resolution. However, manual setting of the grid resolution and thus many trials would be required as to produce the desired effect. Lin et al. show a grid resolution comparison and let the user tune the parameter [2].

Our adaptive grid tries to identify regions where we can set higher or lower resolutions in order to achieve a high quality warping along with a high computational efficiency. Fig. 2 shows these characteristics of using uniform grids with varying resolutions and our adaptive resolution in the application of mental maps. Compared to the quality of the high resolution grid (Fig. 2d), our adaptive grid structure (Fig. 2b) produces the same overall shape with less edges and errors while performing faster. This pattern scales as shown in Fig. 2e: a slower relative growth rate in computational time. We now discuss how to construct our adaptive grids and its optimization.

3.1 Non-uniform Grid Construction

We construct our grid from user defined dimensions by a single root quad by default or by multiple quads if necessary. For each quad, we identify whether the area under the quad is significant by checking for 1) its enclosed road vertex

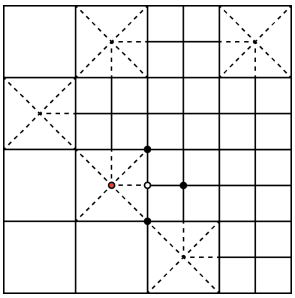


Figure 1: A non-uniform grid with support edges (dashed).

count is greater than c, or 2) inclusion of a special vertex (e.g. guide pattern vertices, which are shown in the inset of Fig. 2a). When either one of criteria satisfies, we divide the quad into four sub-quads, resulting in finer subdivision.

Recursive subdivision can result in an unbalanced quadtree. In our experiments, we observe that the overall shape quality can be improved by balancing the tree through conversion into a restricted grid, wherein neighbors of a quad must be within one level of each other in terms of the tree depth. This restriction prevents a sudden change of deformation rate over a surface, resulting in artifacts minimized [4].

In the uniform grid approach, quads would deform without shape collapsing, because the surrounding quads have the same size. However, in a non-uniform grid like in a quadtree, quads do not necessarily have the same sizes, producing T-junctions that can cause "cracks". Inspired by techniques from terrain rendering, we apply triangulation and provide support edges to remove such cracks. Our goal is to decrease the number of edges and thus increase the optimization performance, while maintaining the accuracy of map deformation. Following the behaviour of the uniform grid, a neighborhood of same sized quads would support themselves. We therefore apply triangulation support edges only to quads with smaller neighbors (Fig. 1). The support edges are included to a quad's edges and are considered to maintain the shape of quads based on Eq. (3).

Smoothing. Laplacian smoothing is applied to each vertex with its orthogonal neighbor vertices. In the case of non-uniform grids, T-junctions may occur due to non-uniform subdivision. To complete its 4-way neighborhood, we consider the centroid vertex, e.g., the red vertex shown in Fig. 1, of the adjacent bigger quad, to serve as a fourth neighbor to these T-junctions.

While useful in maintaining rigidity and implicitly avoiding overlaps, smoothing comes with a problem known as the shrinking, brought to by boundary vertices converging to the centroid of its neighbors towards its internal vertex neighbor. We address this by applying a simple method of Taubin [5] by projecting the Laplacian to the vertex normal.

We encode the system of linear equations from Eq. 5 in the form $\mathbf{Ax} = \mathbf{b}$, and solve it by the conjugate gradient method. Furthermore, we implement it in GPU. Overall, our GPU implementation shows up to 88% increase in performance in our tested settings over CPU. Its detailed discussions are available in the supplementary report.

4 Applications

Our proposed structure can be applied easily in grid based deformation techniques and we demonstrate three such cases. To show different behaviors of uniform grids and our adaptive grids, we show the performance time graph as a function of the residue of each method. A detailed manner of computing residues of different methods and time complexity of our method is available in the sup. report.

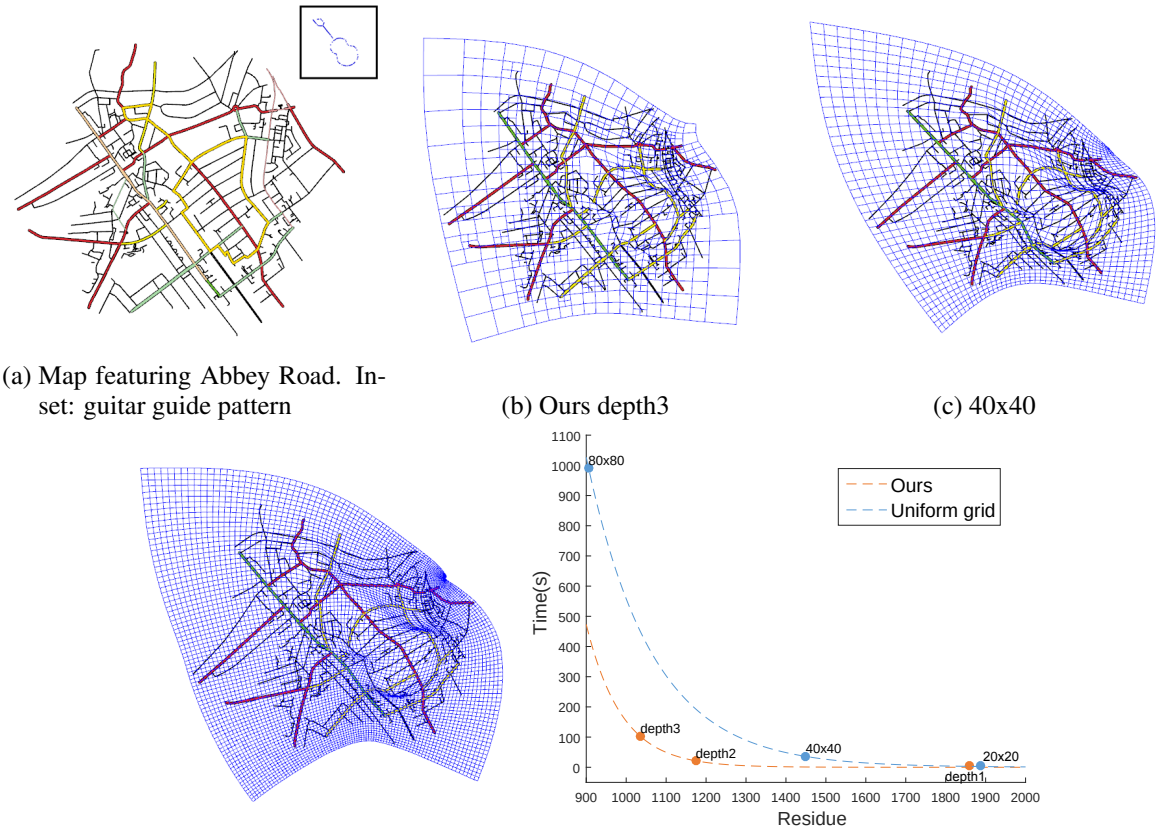
4.1 Mental Maps

Mental maps are defined as maps that feature a pattern formed by roads to help users form a mental image of the location. The pattern is supplied as target vertices that have corresponding real road vertices in the input map. The optimization is guided by those input vertices. Lin et al. [2] designed a uniform grid based deformation to create such maps.

This approach adopts an overlaid grid, while preserving road and quad shapes. The road edge and quad shape preservation objectives are defined as in Eq. 2 and 3. In addition, a guide objective, D_{GV}, is enabled to direct specified road vertices to the input vertex pattern of a mental map, and is shown in below:

$$D_{GV} = \sum_{G \in \mathbf{G}} \left\| \tilde{R} - G \right\|^2, \tag{6}$$

where G is a vertex in the set of target vertices, R is its corresponding road vertex, and $\tilde{R}$ is the



(a) Map featuring Abbey Road. Inset: guitar guide pattern

(b) Ours depth3

(c) 40x40

(d) 80x80

(e) Performance graph.

Figure 2: Abbey road mental map. As the resolution increases, the residue decreases, as shown in (e). Also, our adaptive grid's computational time shows a slower growth rate than the uniform grid. The base grid for our adaptive approach is 10x10. Three different points labeled as depth1, depth2 and depth3 indicate pairs of running time and residue for our method.

deformed version of R. In our method, we replace the uniform grid into our non-uniform grid and apply the optimization process.

As an example, Fig. 2 shows the famous Abbey road having the same name of one of the Beatles' albums. In this example, we want to use the guitar pattern (the inset of Fig. 2a) as a mental map to form the shape of the road. We compare our work to a uniform grid and measure their performance (Fig. 2e). In our tests, we found that as the resolution increases and the residue decreases, the computation time of our adaptive grid has a slower relative growth rate than that of the uniform grid.

As two other applications of our method, we apply our method to road deformation and focus region maps. We also achieve similar improvements to what we have observed in the first application. Details about these two additional applications are available in the supplementary report.

Acknowledgements.

This work was supported in part by MI/KEIT 10070171 and DAPA/DITC (UC160003D) project.

References

[1] J. Bottger, U. Brandes, O. Deussen, and H. Ziezold, "Map warping for the annotation of metro maps," *IEEE CG and A*, vol. 28, 2008.

[2] S.-S. Lin, C.-H. Lin, Y.-J. Hu, and T.-Y. Lee, "Drawing road networks with mental maps," *IEEE TVCG*, 2014.

[3] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, "Laplacian mesh optimization," in *GRAPHITE*, 2006.

[4] B. Von Herzen and A. H. Barr, "Accurate triangulations of deformed, intersecting surfaces," SIGGRAPH, 1987.

[5] G. Taubin, "Linear anisotropic mesh filtering," *Res. Rep. RC2213 IBM*, vol. 1, p. 4, 2001.