Documentation of fast 3D Node generation with variable density in Matlab

Tim Michaels and Alex Vlasiuk

1 Overview

SWe developed the following algorithm for generating nodes with variable density in the cube $\mathcal{I} = [0,1]^3$: Suppose a function $r : \mathcal{I} \to (0,1]$ prescribing the local radius of nodes is given.

1. Partition $\mathcal{I}$ into n_C^3 subcubes of side length $1/n_C$. The choice of n_C roughly corresponds to the resolution of the final node set. Let N_{Max} be the maximum number of nodes in each subcube in the inital distribution. The radius function, n_C, and N_{Max} can be scaled to change the total number of desired nodes N. Experimentally, choosing $N_{Max} = 20$ and n_C such that there are approximately $N/10$ total cubes seems to be a good choice.

2. On a subcube A, let r_A be the average value of r evaluated at the corners of A. This determines N_A, the number of points in A by

$$N_A := \lfloor N_{Max}(1 - r_A) \rfloor .$$

Then N_A points are generated on $\mathcal{I}$ with the following irrationally rotated lattice and scaled and translated to A:

$$\left(\frac{i}{N_A}, \left\{ \alpha \frac{i}{N_A} \right\}, \left\{ \beta \frac{i}{N_A} \right\} \right) i = 1 \dots N_A, \qquad \alpha, \beta \in \mathbb{R} \setminus \mathbb{Q}$$

where $\{x\} := x - \lfloor x \rfloor$ denotes the fractional part of x. While any values of α and β will give an equidistributed lattice as N_A grows, the particular values can be adjusted to provide better distribution for small N_A. We currently use $\alpha = \sqrt{2}, \beta = \sqrt{3}$.

3. Electrostatically repel the points in m steps using the k nearest neighbors of each point. The current implementation has $k = 15$ and the number of repel steps can be adjusted based on desired speed. For the i^{th} repel step, given a point y with nearest neighbors $\{x_j\}_{j=1}^{k}$ form the weighted vector sum

$$x = \sum_{j=1}^{k} \frac{y - x_j}{|y - x_j|^s}$$

for some $s > 3$ and move y away from the direction of x. That is,

$$y \mapsto y - \frac{\delta_y}{(i+1)} \frac{x}{|x|}$$

where δ_y denotes the nearest neighbor distance to y. The kNN tree is not recomputed for each repel step.

This algorithm generates ??? nodes in ??? seconds and (1 million nodes) in ??? seconds.

[Insert Trui pictures]

2 Matlab code

The Matlab files for this code are available from *https://github.com/OVlasiuk/3dRBFnodes*

The main Matlab file is

3 Future Work

Several outstanding issues remain.

- The generation of the lattice on each subcube should be improved. We are working on transforming the lattice so that the density varies linearly within each subcube. This would make the density of the initial point generation more accurate and fewer repulsion steps would be needed.
- The current code does not handle general boundaries beside the cube. Ideally the node distribution restricted to the boundary should be a reasonable 2D node distribution of variable density.
- We are working on parallelizing the repulsion steps and implementing the algorithm on a GPU.