A Generalized 2D and 3D Hilbert Curve

Jakub Červený[1], Zzyv Zzyzek[2]

Abstract
The two and three dimensional Hilbert curves are fractal space filling curve that map the unit interval to the unit square or unit cube. Hilbert curves preserve locality, keeping distance from the unit interval in their respective higher dimensional embedding and, for this reason, find applications in scene rendering to job scheduling. Finite approximations of the Hilbert curve can be constructed in stages by stopping the recursive construction at a specified depth. One limitation of the Hilbert curve approximation construction is that the regions need to be exact powers of two, making it difficult to work in many real world scenarios. In this paper, we present the Gilbert curve, a conceptually straight forward generalization of the Hilbert curve that works on arbitrary rectangular regions. The construction provides reasonable worst case run-time guarantees for random access forward and back lookup from the unit interval to the embedded 2 or 3 dimensional space. We also provide comparisons to other Hilbert curve generalization methods and investigate overall quality metrics of the Gilbert curve construction.

1. Introduction

1.1. Overview

We present a generalized Hilbert curve for 2D and 3D curves that works on arbitrary rectangular regions. We call our realization of the generalized Hilbert curve a *Gilbert curve*.

An overview of the benefits of the Gilbert curve are that it is:

- Valid on arbitrary rectangular regions
- Equivalent to the Hilbert curve when dimensions are exact powers of 2
- Efficient at random access lookup
- Algorithmically conceptually straight forward
- Able to realize curves without notches unnecessarily and limits the resulting curve to a single notch when forced
- Similar in measures of locality to the Hilbert curve

Some drawbacks are that:

- Our implementation involves a recursive solution for random access lookup which might be too slow for applications that require better than $O(\lg N)$ runtime and memory usage
- Might not adequately capture some features of a Hilbert curve

Further, we show:

- Measures of locality are preserved (Section X)
- Trivial extensions to create generalized Moore curves (*Gore curve*, Section X)
- Other metrics to quantify qualities of the curve and their comparison to some other space filling curves

Generalizations of the Hilbert curve to non power of two rectangular cuboid regions has been explored before but are overly complicated algorithmically, create unbalanced curves and often don't generalize well to 3D. Our realization focuses on algorithmic conceptual simplicity, provides balanced resulting curves and works in both 2D and 3D.

Space filling curves are a specialization of a more general Hamiltonian path, but have a more stringent requirement of local connectivity. The local connectivity, or locality, preserves a measure of nearness, where points from the source unit line are near in the embedded space.

The local connectivity requirements preclude things like *zig-zag* Hamiltonian paths, as nearby points in the embedded dimension can be far from the origin line.

The Gilbert curve algorithm works by choosing sub rectangular cuboid regions, or blocks, to recursively find a connecting path. During the course of subdivision, if the cuboid block regions stray too far from being cube like, a simpler subdivision is done to try and bring the subdivision closer to being cube like. The subdivision is done until a base case is reached and the lowest unit of the curve can be realized.

For a specified path start and end position with certain side dimensions, a Hamiltonian path is not always possible. In such a case, the Gilbert curve will introduce a single diagonal path move, called a *notch*.

1.2. Definitions

We concern ourselves with a space filling curve, ω, through a rectangular region (N_x, N_y, N_z), $N = (N_x \cdot N_y \cdot N_z)$:

$$\begin{aligned} k \in & \quad \{0 \dots (N-1)\}, \\ & \quad x_k, y_k, z_k \in \mathbb{N} \\ \omega_k = & \quad (x_k, y_k, z_k) \\ \omega = & \quad (\omega_0, \omega_1, \dots, \omega_{N-1}) \\ k > 1 & \quad \rightarrow |\omega_{k-1} - \omega_k| = 1 \\ \forall i, j \in & \quad \{0 \dots (N-1)\}, i \neq j \\ \rightarrow & \quad \omega_i \neq \omega_j \end{aligned}$$

2. Related Work

3. Algorithm

4. Experiments and Results

5. Conclusion

References

A. Appendix

https://github.com/jakubcerveny (J. Červený);
https://zzyzek.github.io/ (Z. Zzyzek)
0009-0005-2175-1166 (Z. Zzyzek)

© 2025 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).