A Generalized 2D and 3D Hilbert Curve

Jakub Červený and Zzyv Zzyzek

Abstract

The two and three dimensional Hilbert curves are fractal space filling curves that map the unit interval to the unit square or unit cube while preserving a notion of closeness, or locality, after the map. We present the Gilbert curve, a conceptually straight forward generalization of the Hilbert curve, that works on arbitrary rectangular regions, overcoming the limitation of the Hilbert curve that requires side lengths be exact powers of two. Our construction of the Gilbert curve provides a notion of *harmony* that provide alternate subdivision schemes when a side length is much larger than the rest, creating more visual pleasing curves as a result.

Generalized Hilbert Curves

In a website application and scanned note [1], Tautenhahn provided the basis for a generalized 2D space filling curve. Tautenhahn also included policies for when to subdivide regions preferentially in only one dimension that help to create more *harmonious* curve realizations. Tautenhahn's exploration details the parity arguments necessary for when a subdivision scheme can be employed without creating diagonal moves, which we call *notches* here.

In this paper, we extend Tautenhahn's ideas to create a 2D and 3D generalized Hilbert curve. We further extend Tautenhahn's core ideas on when to use alternate subdivision schemes when one length is much larger than the rest, what we call *eccentric cases*, and apply them to a 3D generalized Hilbert curve. Tautenhahn's unpublished reasoning behind the constants used in the 2d eccentric split case are briefly discussed later[1].

.

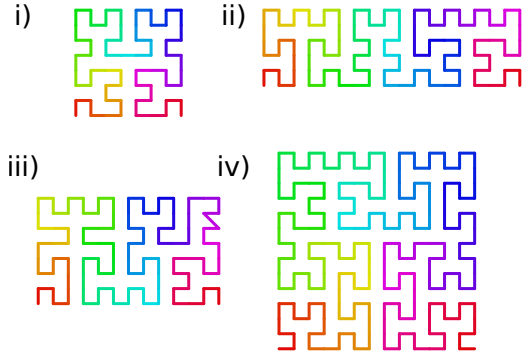


Figure 1: *2D Gilbert curves for i)* 8×8*, ii)* 18×6*, iv)* 13×8 *(with notch), iv)* 14×14

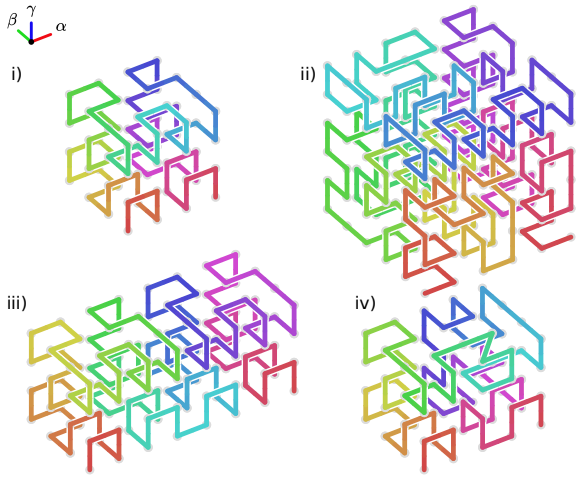


Figure 2: *3D Gilbert curves for i)* $4 \times 4 \times 4$*, ii)* $6 \times 6 \times 6$*, iii)* $8 \times 4 \times 4$*, iv)* $5 \times 4 \times 4$ *(with notch)*

[1]Through personal communication with the authors, Tautenhahn kindly provided the reasoning for the constants used in the eccentric split

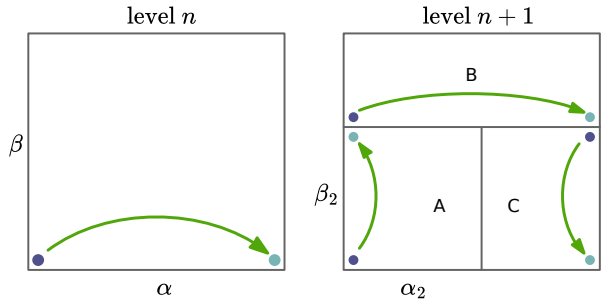


Figure 3: *Subdivision strategy for the 2D Gilbert curve.*

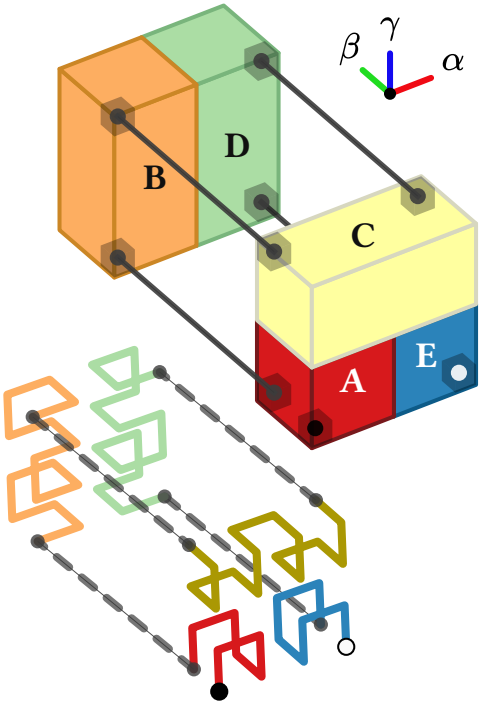


Figure 4: *The main subdivision strategy for the 3D Gilbert curve.*

Subdivision Strategies

For both the 2D and 3D Gilbert curve, a subdivision template is chosen to recursively partition the region. Figure 3 shows the 2D template and figure 4 shows the 3D template.

The vectors $\alpha, \beta, \gamma \in \mathbb{Z}^3$ provide generalized notions of width (α), height (β), and depth (γ), so that we can transform rectangular or cuboid regions as necessary for the recursion. Each of α, β, γ will only have one non-zero component and will be orthogonal to each other, representing a current snapshot of the current frame of reference.

When dividing into sub-regions, side lengths are chosen to be integral and with even side lengths preferred for the first subdivided region (region *A* in figures 3 and 4). A path is recursively chosen for each of the subregions and, after resolution, endpoints are connected. Figure 5 shows the choice in 2D of even or odd regions depending on side length parity.

When regions have a side length that is much larger than the rest, another subdivision scheme is used. We call these cases *eccentric subdivisions*, as the larger side length makes the region lopsided.

If one side length is much larger than the rest, both the 2D and 3D Gilbert curve split the region in two. For the 3D case, if one side length is much smaller than the rest but with the remaining lengths roughly equal, the region is subdivided as if it were a 2D Gilbert curve, taking the smaller side length as the "flat" dimension.

The eccentric subdivision schemes provide a more visually pleasing solution as regimented subdivisions might take paths that are long and skinny or wide and squat. In the next section, we will briefly go over motivations for the threshold values of when to use the eccentric subdivision schemes.

For the 2D Gilbert curve, if $|\alpha|$ is odd, there will be a diagonal move, or *notch*, for the resulting path. The subdivision scheme will keep the notch in a single region, limiting the notch count to one.

For the 3D Gilbert curve, if any of the subdivided regions has endpoints that lay in a direction with odd length, a notch will appear. This means the number of notches for the 3D Gilbert curve isn't limited to one.

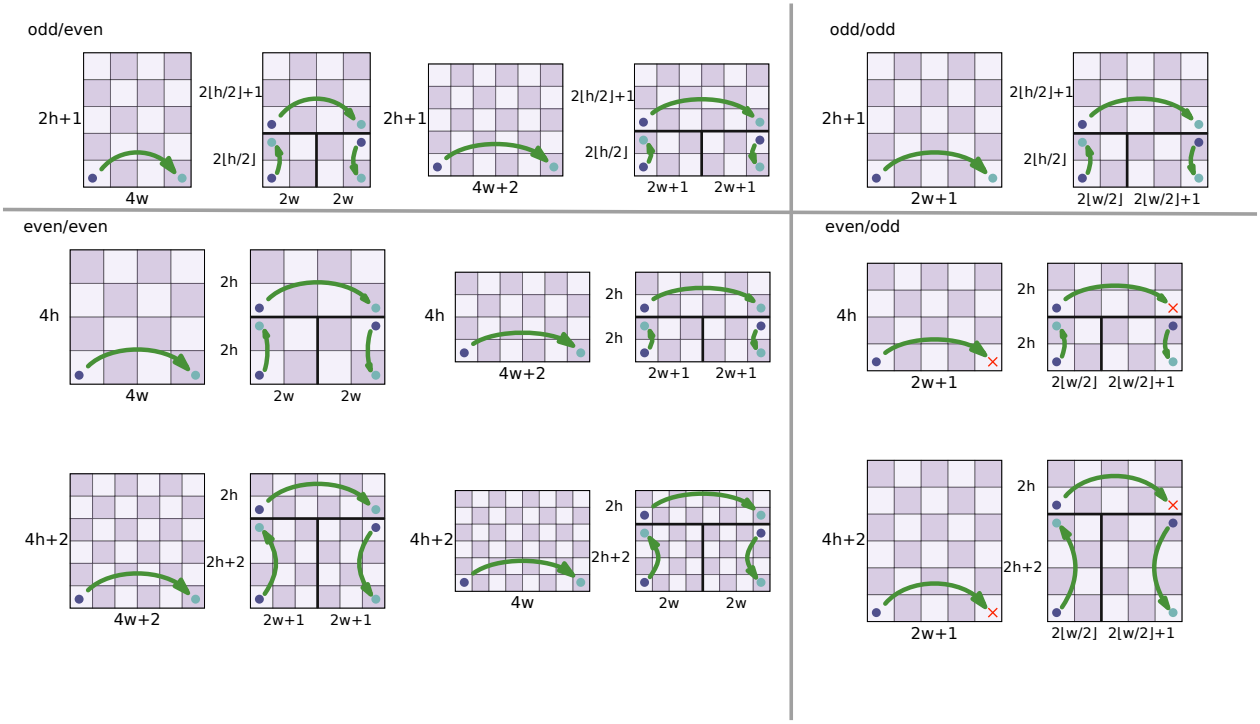


Figure 5: *Enumeration of the subdivision template depending on different parities of α and β dimensions.*

Eccentric Subdivision

When side lengths become too lopsided, an alternate *eccentric* subdivision scheme is chosen to split along the larger side length.

For each eccentric subdivision, a proportion threshold needs to be chosen to determine when a side lengths become too lopsided to warrant an alternate subdivision. Here we motivate the eccentric subdivision constants for the 2D case.

Consider a *defect* function, $\lambda_2 : \mathbb{N}^2 \mapsto \mathbb{N}$, that measures the area relative to what the area would be if it just the minimum side length were taken:

$$\lambda_2(|\alpha|, |\beta|) = \frac{|\alpha| \cdot |\beta|}{\min(|\alpha|, |\beta|)^2}$$

If there is a disjoint subdivision of a volume V_0 to $V_1 = (V_{0,0}, V_{0,1}, \ldots, V_{0,m-1})$, $(V_0 = \cup_k V_{0,k})$, define the subdivided volume's *average defect* to be the sum of defects weighted by their proportional volume:

$$\lambda_s(V_1) = \sum_k \frac{\mathrm{Vol}(V_{0,k})}{\mathrm{Vol}(V_0)} \cdot \lambda_2(V_{0,k})$$

The defect gives a coarse idea of how lopsided or *eccentric* a region is. If the defect is too high, we might want to split the larger sides while keeping the smaller sides the same size. Reducing the average defect attempts to make each subdivided region more square-like. When subdivided areas are more square-like, we say that the subdivided regions are more *harmonious*.

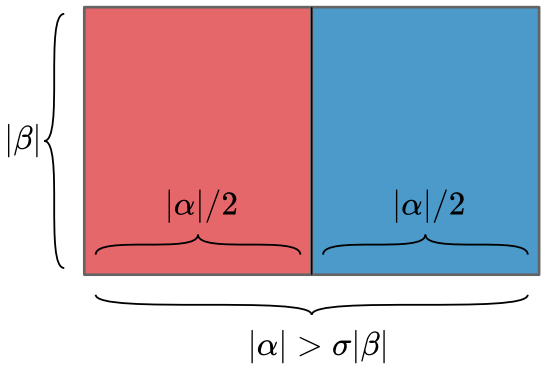


Figure 6: *When the length of the width-like dimension ($|\alpha|$) gets larger than a threshold of the height-like dimension ($\sigma|\beta|$), then the width like dimension is split into two.*

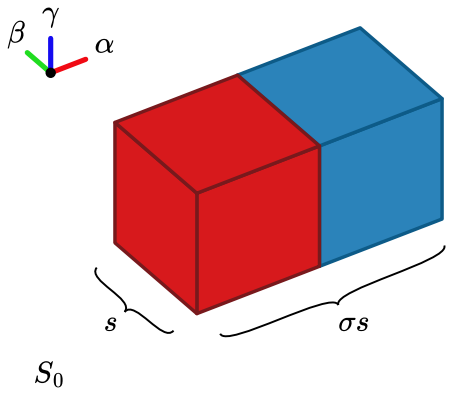


Figure 7: *An S_0 eccentric subdivision showing the threshold, σ, used to determine when we should use this split.*

If the α side length is significantly longer than the β side length, we want to subdivide the rectangle into two nearly equal regions and recursively find a Gilbert curve in each region. We will justify the constant $(3/2)$ as the ratio threshold to split on, using an argument originally developed by L. Tautenhahn [1].

The defect of a rectangle with side length $|\alpha|$ and $|\beta|$, assuming $|\alpha| > |\beta|$, is $(|\alpha|/|\beta|)$. After subdivision, if we assume $|\alpha| < 2|\beta|$, the resulting defect is $2|\beta|/|\alpha|$. For a defect reduction, we search for conditions in which the ratio of the resulting defect to the original defect is less than unity, giving the equation $|\alpha|/|\beta| > \sqrt{2}$.

Since $\sqrt{2} \approx 1.4142 < (3/2)$, if we choose $|\alpha|/|\beta| > (3/2)$ we can be assured a defect reduction. In the case when $|\alpha| > 2|\beta|$, it is easy to verify that the defect is always reduced [2].

Defect reductions for the 3D case can be done, but are more complicated. The constants for the 3D subdivision in this paper were observed to yield visually pleasing results without concern for defect optimization.

[1] Given to us through personal communication with L. Tautenhahn

[2] The ratio of the defects before and after the split is $(1/2)$

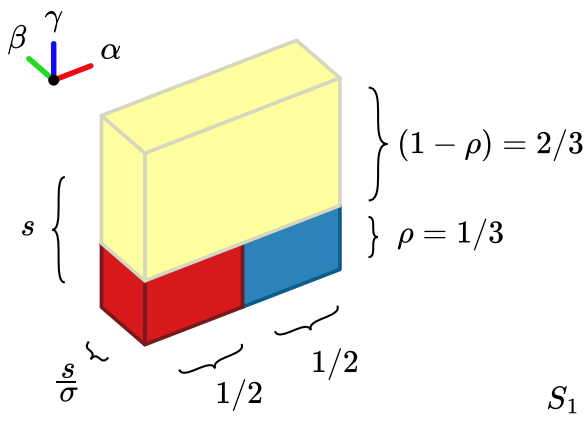


Figure 8: *An S_1 eccentric subdivision showing the threshold, σ, used to determine when we should use this split and the split point ratio, ρ, to reduce the defect for a more harmonious subdivision.*

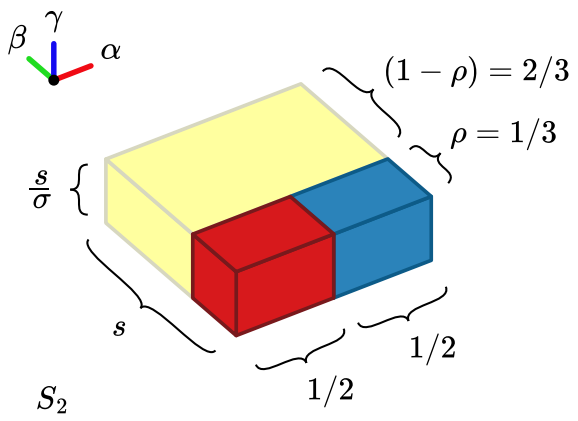


Figure 9: *An S_1 eccentric subdivision showing the threshold, σ, used to determine when we should use this split and the split point ratio, ρ, to reduce the defect for a more harmonious subdivision.*

Algorithm

Algorithm 1

```
# p, α, β ∈ ℤ³
function GILBERT2D(p, α, β)
    α₂, β₂ = div(α, 2), div(β, 2)
    if (|β| ≡ 1) then
        yield p + i · δ(α) forall i ∈ |α|
    else if (|α| ≡ 1) then
        yield p + i · δ(α) forall i ∈ |β|
    else if (2|α| > 3|β|) then
        if (|α₂| > 2) and
           (|α₂| mod 2 ≡ 1) then
            α₂ ← α₂ + δ(α)
        end if
        yield Gilbert2D(p, α₂, β)
        yield Gilbert2D(p + α₂, α − α₂, β)
    else
        if (|β₂| > 2) and
           (|β₂| mod 2 ≡ 1) then
            β₂ ← β₂ + δ(β)
        end if
        yield Gilbert2D(p,
                        β₂, α₂)
        yield Gilbert2D(p + β₂,
```

Algorithm 2

```
# p, α, β, γ ∈ ℤ³
function GILBERT3D(p, α, β, γ)
    return Gilbert2D(p,β,γ) if (|α| ≡ 1)
    return Gilbert2D(p,α,γ) if (|β| ≡ 1)
    return Gilbert2D(p,α,β) if (|γ| ≡ 1)
    α₂ ← div(α, 2) + Δ(α₂, α)
    β₂ ← div(β, 2) + Δ(β₂, β)
    γ₂ ← div(γ, 2) + Δ(γ₂, γ)
    if (2|α| > 3|β|) and (2|α| > 3|γ|))
       yield Gilbert3D(p,α₂,β,γ)
       yield Gilbert3D(p + α₂,α − α₂,β,γ)
    if (3|β| > 4|γ|)
       yield Gilbert3D(p,β₂,γ,α₂)
       yield Gilbert3D(p + β₂,α,β − β₂,γ)
       yield Gilbert3D(p+
                       (α − δ(α))+
                       (β₂ − δ(β)),
                       −β₂,γ,−(α − α₂))
    if (3|γ| > 4|β|)
       yield Gilbert3D(p,γ₂,α₂,β)
       yield Gilbert3D(p + γ₂,α, β, γ − γ₂)
       yield Gilbert3D(p+
                       (α − δ(α))
                       (γ2 − δ(γ)),
                       −γ₂,−(α − α₂), β)

    else (3|β| > 4|γ|)
       yield Gilbert3D(p,β₂,γ₂,α₂)
       yield Gilbert3D(p + β₂,γ,α₂,β₂)
       yield Gilbert3D(p+
```

Algorithm 1 shows the pseudo-code for computing the 2D Gilbert curve. Note that α and β are taken to be vectors in 3D, where the third dimension can be ignored if a purely 2D curve is desired. The generalization to 3D allows the Gilbert2D function to be used unaltered when the 3D Gilbert curve needs to trace out in-plane sub-curves.

Algorithm 1 assumes standard Euclidean two norm ($|v| = \sqrt{v_0^2 + v_1^2 + v_2^2}$) and abuses notation by allowing scalar to vector multiplication ($i \in \mathbb{Z}, v \in \mathbb{Z}^3, i \cdot v \rightarrow (i \cdot v_0, i \cdot v_1, i \cdot v_2)$). where the context is clear.

The $\delta(\cdot)$ function returns one of the six directional vectors indicating which of the major signed axis aligned directions the input vector points to $((\pm 1, 0, 0), (0, \pm 1, 0), (0, 0, \pm 1))$. To make things more concise, a convenience function is used, $(\Delta(\rho_2, \rho) \stackrel{\text{def}}{=} \delta(\rho)$ **if** $|\rho_2| \equiv 1 \bmod 2$ and $|\rho| > 2)$, that prefers even lengths in the non-degenerate case.

Conclusion and Future Directions

The Gilbert curve presented in this paper provides one intepretation for a generalized Hilbert curve in 2D and 3D to arbitrary side lengths. Generaliing a Hilbert space filling curve to arbitrary side lengths is not a well posed problem and is open to interpretation, and the solution presented here is one such adaptation.

When desinging an algorithm to generalize the Hilbert curve, four features that might be desirable are:

Hilbert-esque	Reduces to a Hilbert curve when side lengths are equal and are exact powers of two
Stabilty	Curve realizations converge as the cuboid side lengths increase proportionally
Harmony	Realizations use eccentric splits to reduce the *defect*
Notch-Limited	Curve realizations are notch free unless a single notch is necessary

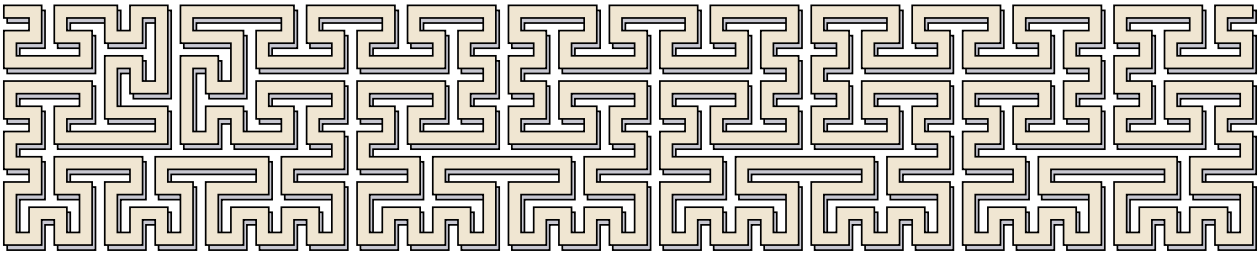
The 2D Gilbert curve presented here has all of these properties while the 3D Gilbert curve lacks harmony and the notch-limited feature.

Future work could focus on creating 3D Gilbert curve that has all of these features, perhaps by relaxing strict endpoint requirements at the cuboid corners. Some other natural ideas are to extend the concepts above to other space filling curves or ciruits, or, more generally, to try and create an arbitrary space filling curve in a cuboid region with arbitrary endpoints.

A libre/free implementation for the Gilbert curve in 2D and 3D has been developed and can be downloaded from its repository [3].

References

[1] L. Tautenhahn. “Draw a space-filling curve of arbitrary size.” 2003. https://lutanho.net/pic2html/draw_sfc.html.



[3] https://github.com/jakubcerveny/gilbert