

# A Generalized 2D and 3D Hilbert Curve

Jakub Červený, 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. Hilbert curves preserve locality, keeping distance from the unit interval to their respective higher dimensional embeddings. Finite approximations of the Hilbert curve can be constructed in stages by stopping the recursive construction at a specified depth, providing locality from a discrete array of points to points in the embedded dimension. The locality property of the Hilbert curve can help with caching optimizations based on order and finds uses as a companion method in applications ranging from job scheduling to scene rendering. One limitation of the Hilbert curve approximation construction is that the regions need to be exact powers of two, making it difficult to work with 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 lookups for both two and three dimensions. 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 the *Gilbert curve*, a generalized Hilbert curve for 2D and 3D, that works on arbitrary rectangular regions.

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 lookups ( $O(\lg N)$ )
- A conceptually straight forward algorithm
- 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 is recursive, which may undesirable for some 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 (Section X)
- Comparison metrics to other space filling curves (Section X)

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 a conceptually simple algorithm which creates pleasing 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 remain close 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. If one extent of the cuboid becomes too large or small relative to the other lengths, a special subdivision is performed to try make further subdivisions more cube-like. Subdivisions are performed until a base case is reached and the lowest unit of the curve can be realized.

A Hamiltonian path is not always realizable for certain side lengths and endpoint constraints. 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,  $\omega$ , through a rectangular region  $(N_x, N_y, N_z)$ ,  $N = (N_x \cdot N_y \cdot N_z)$ :

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

## 2. Related Work

## 3. Algorithm

### 3.1. Overview

For both the 2D and 3D generalized Hilbert (Gilbert) curve, a template is chosen to recursively subdivide the area or volume. When recursively descending the sub-volumes, we will need to translate to a new reference point and rotate by increments of  $(\pi/2)$  radians, maintaining cuboid regions that are axis-aligned. We use the convention of labeling  $\alpha, \beta, \gamma \in \mathbb{Z}^3$  for the width-like, height-like and depth-like dimension vectors, respectively.

Each of  $\alpha, \beta$  and  $\gamma$  start off as axis-aligned vectors of integral length and are only rotated by units of  $(\pi/2)$  radians, remaining axis aligned throughout.  $\alpha, \beta$  and  $\gamma$  represent

the dimensions of the sub-cuboid region and the local basis when tracing out a curve.

The next sub-section discusses parity arguments for when a valid curve can be traced out in a sub-region. We then discuss the 2D Gilbert curve and end with the 3D Gilbert curve.

When talking about the 2D Gilbert curves, we assume vectors are in  $\mathbb{Z}^3$  so they can be used in the more general 3D.

### 3.2. Valid Paths from Grid Parity

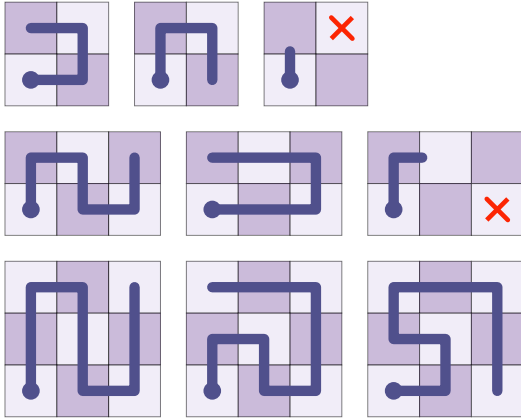
The feasibility of determining whether there exists a Hamiltonian path in a rectangular cuboid grid region can be accomplished through parity arguments. Label grid cell points in a volume as 0 or 1, alternating between labels with every axis-aligned single step move. Any Hamiltonian path that ends at one of the three remaining corners has to have the same parity as the starting point if the volume is odd, or different parity if the volume is even.

Path Possible	Volume	
	even	odd
$ \alpha $ even	Yes	Yes
$ \alpha $ odd	No	Yes

**Table 1**

Table showing when a Hamiltonian cycle is possible. Here,  $|\alpha|$  is the absolute difference in start and end position of the path which coincides with one of the axis aligned side length of the cuboid volume. A Hamiltonian path is not possible on only in the case when  $|\alpha|$  is odd and the volume is even.

For a path starting at  $(0, 0, 0)$  and ending at  $|\alpha| \in \mathbb{N}$  away in one of the axis-aligned dimensions, then Table 1 enumerates this condition under which a valid path is possible. Figure 1 illustrates this for starting position  $(0, 0)$  with volumes  $(2 \times 2)$ ,  $(3 \times 2)$  and  $(3 \times 3)$ , where a red cross indicating a precluded endpoint.



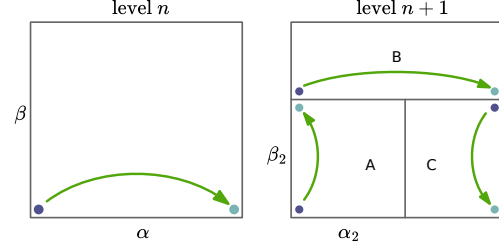
**Figure 1:** Illustrative examples of Hamiltonian paths height/width that are even/even, even/odd and odd/odd, respectively, when starting from the lower left hand corner

Without loss of generality, we will assume a curve starts from position  $p_s = (0, 0, 0)$  and has proposed endpoint at  $p_e = ((w-1), 0, 0)$ , with a cuboid region as  $\alpha = (w, 0, 0)$ ,  $\beta = (0, h, 0)$ ,  $\gamma = (0, 0, d)$ . We know a Hamiltonian path is precluded if  $(|\beta| \cdot |\gamma| \bmod 2 \neq 1)$ . That is, a

Hamiltonian from  $p_s$  to  $p_e$  is precluded when  $\alpha$  is odd and at least one of  $|\beta|$  or  $|\gamma|$  is even.

We will show that a Hamiltonian path is possible from  $p_s$  to  $p_e$  when  $|\alpha|$  is even or when  $|\alpha|$ ,  $|\beta|$  and  $|\gamma|$  are all odd. **TODO!!!**

### 3.3. 2D Generalized Hilbert Function (Gilbert2D)



**Figure 2:** For the 2D Gilbert curve, the main rectangle is subdivided into three sub regions, making sure their paths connect and preferring even side lengths for the first rectangular subdivision. Partitioning the rectangle in this way is what we call a *U-split*.

For the 2D Gilbert curve, a *U-split* template is used, highlighted in figure 2. The *U-split* breaks the region into three sub-blocks, labeled *A*, *B* and *C*, where the *A* is chosen to be half the width like dimension,  $\alpha$ , and half the height like dimension,  $\beta$ , with a preference to make the height like dimension,  $\beta$ , of even length.

3.

Algorithm 1 shows the pseudo-code for computing the 2D Gilbert curve. Note that  $\alpha$  and  $\beta$  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.

### 3.4. 3D Generalized Hilbert Function (Gilbert3D)

## 4. Experiments and Results

## 5. Conclusion

## References

### A. Appendix

#### A.1. Defect

Call the *defect* a function  $\lambda_d : \mathbb{N}^d \mapsto \mathbb{N}$ :

$$\lambda_2(w, h) = \frac{w \cdot h}{\min(w, h)^2}$$

$$\lambda_3(w, h, d) = \frac{w \cdot h \cdot d}{\min(w, h, d)^3}$$

If there is a disjoint subdivision of a volume  $V_0$  to  $V_1 = (V_{0,0}, V_{0,1}, \dots, V_{0,m-1})$ ,  $V_0 = \cup_k V_{0,k}$ , define the *average defect* of the subdivided volume to be:

$$\lambda_s(V_1) = \sum_k \frac{\text{vol}(V_{0,k})}{\text{Vol}(V_0)} \cdot \lambda(V_{0,k})$$

This weights the defect of each subdivided cuboid by its proportional volume.



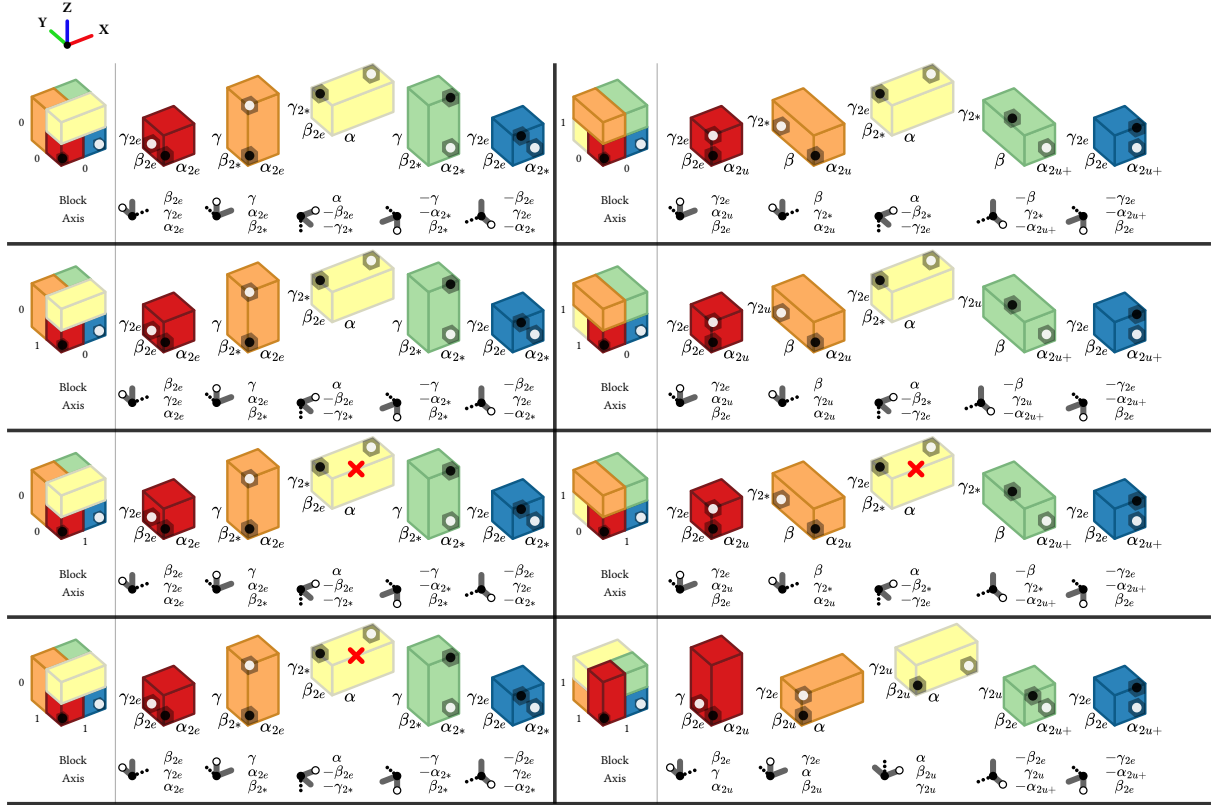


Figure 4: Bulk recursion J-split atlas for the 3D Gilbert algorithm

A.4.  $h \gg w \sim d$

---

**Algorithm 1** 2D Generalized Hilbert Function (Gilbert2D)

---

```

#  $p, \alpha, \beta \in \mathbb{Z}^3$ 
function GILBERT2D( $p, \alpha, \beta$ )

     $\alpha_2, \beta_2 = (\alpha//2), (\beta//2)$ 

    if ( $|\alpha| \equiv 1$ ) then
        yield  $p + i \cdot \delta(\alpha)$  forall  $i \in |\alpha|$ 

    else if ( $|\alpha| \equiv 1$ ) then
        yield  $p + i \cdot \delta(\alpha)$  forall  $i \in |\beta|$ 

    else if ( $2|\alpha| > 3|\beta|$ ) then
        if ( $|\alpha_2| > 2$ ) and ( $|\alpha_2| \bmod 2 \equiv 1$ ) then
             $\alpha_2 \leftarrow \alpha_2 + \delta(\alpha)$ 
        end if

        yield Gilbert2D( $p,$ 
                         $\alpha_2, \beta$ )

        yield Gilbert2D( $p + \alpha_2,$ 
                         $\alpha - \alpha_2, \beta$ )

    else
        if ( $|\beta_2| > 2$ ) and ( $|\beta_2| \bmod 2 \equiv 1$ ) then
             $\beta_2 \leftarrow \beta_2 + \delta(\beta)$ 
        end if

        yield Gilbert2D( $p,$ 
                         $\beta_2, \alpha_2$ )

        yield Gilbert2D( $p + \beta_2,$ 
                         $\alpha, (\beta - \beta_2)$ )

        yield Gilbert2D( $p + \alpha - \delta(\alpha) + \beta_2 - \delta(\beta),$ 
                         $\beta_2, -(\alpha - \alpha_2)$ )

    end if
end function

```

---



---

**Procedure 2**  $S_0$ -Split function (eccentric split)

---

```

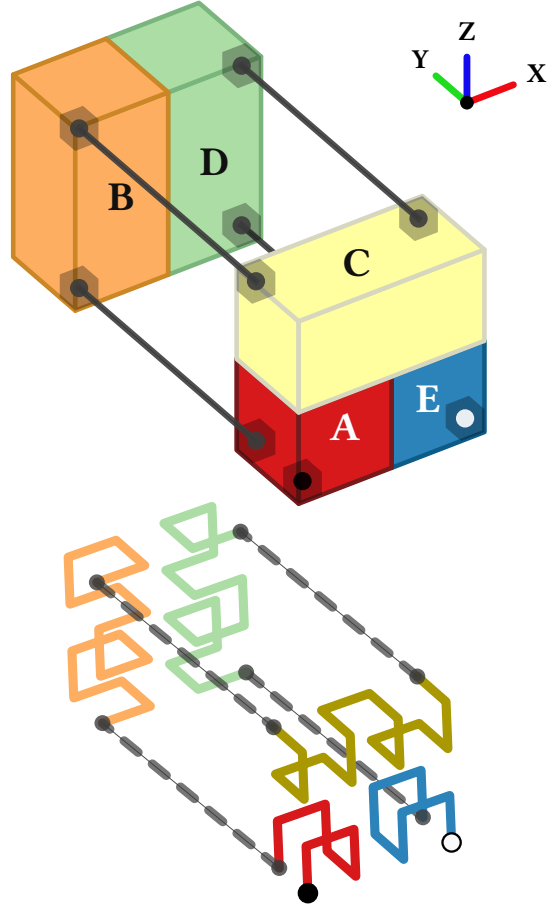
# split halfway on  $\alpha$ 
function  $S_0(p, \alpha, \beta, \gamma)$ 
     $\alpha_2 \leftarrow (\alpha//2)$ 
    if ( $|\alpha| > 2$ ) and ( $|\alpha_2| \bmod 2 \equiv 1$ ) then
         $\alpha_2 \leftarrow \alpha_2 + \delta(\alpha)$ 
    end if

    yield Gilbert3D( $p,$ 
                     $\alpha_2, \beta, \gamma$ )

    yield Gilbert3D( $p + \alpha_2,$ 
                     $(\alpha - \alpha_2), \beta, \gamma$ )
end function

```

---



**Figure 5:** The  $J_0$ -split subdivision, representing the main subdivision of the bulk recursion for the 3D Gilbert curve case.

---

**Procedure 3**  $S_1$ -Split functions (eccentric split)

---

```

# split  $\frac{1}{3}$  on  $\gamma$  and halfway on  $\alpha$ 
function  $S_1(p, \alpha, \beta, \gamma)$ 
     $\alpha_2, \gamma_3 \leftarrow (\alpha//2), (\gamma//3)$ 
    if ( $|\alpha| > 2$ ) and ( $|\alpha_2| \bmod 2 \equiv 1$ ) then
         $\alpha_2 \leftarrow \alpha_2 + \delta(\alpha)$ 
    end if
    if ( $|\gamma| > 2$ ) and ( $|\gamma_3| \bmod 2 \equiv 1$ ) then
         $\gamma_3 \leftarrow \gamma_3 + \delta(\gamma)$ 
    end if

    yield Gilbert3D( $p,$ 
                     $\gamma_3, \alpha_2, \beta$ )

    yield Gilbert3D( $p + \gamma_3,$ 
                     $\alpha, \beta, (\gamma - \gamma_3)$ )

    yield Gilbert3D( $p + \alpha - \delta(\alpha) + \gamma_3 - \delta(\gamma),$ 
                     $\gamma_3, (\alpha - \alpha_2), \beta$ )

end function

```

---

---

**Procedure 4**  $S_2$ -Split function (eccentric split)

---

# split  $\frac{1}{3}$  on  $\beta$  and halfway on  $\alpha$   
**function**  $S_2(p, \alpha, \beta, \gamma)$   
     $\alpha_2, \beta_3 \leftarrow (\alpha//2), (\beta//3)$   
    **if**  $(|\alpha| > 2)$  and  $((|\alpha_2| \bmod 2) \equiv 1)$  **then**  
         $\alpha_2 \leftarrow \alpha_2 + \delta(\alpha)$   
    **end if**  
    **if**  $(|\beta| > 2)$  and  $((|\beta_3| \bmod 2) \equiv 1)$  **then**  
         $\beta_3 \leftarrow \beta_3 + \delta(\beta)$   
    **end if**  
  
    **yield** Gilbert3D( $p,$   
         $\beta_3, \gamma, \alpha_2$ )  
  
    **yield** Gilbert3D( $p + \beta_3,$   
         $\alpha, (\beta - \beta_3), \gamma$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \beta_3 - \delta(\beta),$   
         $-\beta_3, \gamma, -\alpha$ )  
  
**end function**

---

---

**Procedure 5**  $J_0$ -Split function

---

#  $|\gamma|$  even  
**function**  $J_0(p, \alpha, \beta, \gamma)$   
  
     $\alpha_2, \beta_2, \gamma_2 \leftarrow (\alpha//2), (\beta//2), (\gamma//2)$   
  
    # prefer initial block even  
     $\alpha_2 = \alpha_2 + \delta(\alpha)$  **if**  $(|\alpha_2| > 2)$  and  $(|\alpha_2| \bmod 2 \equiv 1)$   
     $\beta_2 = \beta_2 + \delta(\beta)$  **if**  $(|\beta_2| > 2)$  and  $(|\beta_2| \bmod 2 \equiv 1)$   
     $\gamma_2 = \gamma_2 + \delta(\gamma)$  **if**  $(|\gamma_2| > 2)$  and  $(|\gamma_2| \bmod 2 \equiv 1)$   
  
    **yield** Gilbert3D( $p,$   
         $\beta_2, \gamma_2, \alpha_2$ )  
  
    **yield** Gilbert3D( $p + \beta_2,$   
         $\gamma, \alpha_2, \beta - \beta_2$ )  
  
    **yield** Gilbert3D( $p + \beta_2 - \delta(\beta_2) + \gamma - \delta(\gamma),$   
         $\alpha, -\beta_2, -(\gamma - \gamma_2)$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \beta_2 + \gamma - \delta(\gamma),$   
         $-\gamma, -(\alpha - \alpha_2), (\beta - \beta_2)$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \beta_2 - \delta(\beta),$   
         $-\beta_2, \gamma_2, -(\alpha - \alpha_2)$ )  
  
**end function**

---

---

**Procedure 6**  $J_1$ -Split function

---

#  $|\gamma|$  odd, one of  $|\alpha|$  or  $|\beta|$  even  
**function**  $J_1(p, \alpha, \beta, \gamma)$   
  
     $\alpha_2, \beta_2, \gamma_2 \leftarrow (\alpha//2), (\beta//2), (\gamma//2)$   
  
    # prefer  $\beta_2, \gamma_2$  even but force  $\alpha_2$  odd  
     $\alpha_2 = \alpha_2 + \delta(\alpha)$  **if**  $(|\alpha_2| > 2)$  and  $(|\alpha_2| \bmod 2 \equiv 0)$   
     $\beta_2 = \beta_2 + \delta(\beta)$  **if**  $(|\beta_2| > 2)$  and  $(|\beta_2| \bmod 2 \equiv 1)$   
     $\gamma_2 = \gamma_2 + \delta(\gamma)$  **if**  $(|\gamma_2| > 2)$  and  $(|\gamma_2| \bmod 2 \equiv 1)$   
  
    **yield** Gilbert3D( $p,$   
         $\gamma_2, \alpha_2, \beta_2$ )  
  
    **yield** Gilbert3D( $p + \gamma_2,$   
         $\beta, \gamma - \gamma_2, \alpha_2$ )  
  
    **yield** Gilbert3D( $p + \gamma_2 - \delta(\gamma) + \beta - \delta(\beta),$   
         $\alpha, -(\beta - \beta_2), -\gamma_2$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \beta - \delta(\beta) + \gamma_2 - \delta(\gamma),$   
         $\beta, \gamma - \gamma_2, -(\alpha - \alpha_2)$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \gamma_2 - \delta(\gamma),$   
         $-\gamma_2, -(\alpha - \alpha_2), \beta_2$ )  
  
**end function**

---

---

**Procedure 7**  $J_2$ -Split function

---

#  $|\alpha|, |\beta|, |\gamma|$  odd  
**function**  $J_2(p, \alpha, \beta, \gamma)$   
  
     $\alpha_2, \beta_2, \gamma_2 \leftarrow (\alpha//2), (\beta//2), (\gamma//2)$   
  
    # prefer  $\beta_2, \gamma_2$  even but force  $\alpha_2$  odd  
     $\alpha_2 = \alpha_2 + \delta(\alpha)$  **if**  $(|\alpha_2| > 2)$  and  $(|\alpha_2| \bmod 2 \equiv 0)$   
     $\beta_2 = \beta_2 + \delta(\beta)$  **if**  $(|\beta_2| > 2)$  and  $(|\beta_2| \bmod 2 \equiv 1)$   
     $\gamma_2 = \gamma_2 + \delta(\gamma)$  **if**  $(|\gamma_2| > 2)$  and  $(|\gamma_2| \bmod 2 \equiv 1)$   
  
    **yield** Gilbert3D( $p,$   
         $\beta_2, \gamma, \alpha_2$ )  
  
    **yield** Gilbert3D( $p + \beta_2,$   
         $\gamma_2, \alpha, (\beta - \beta_2)$ )  
  
    **yield** Gilbert3D( $p + \beta_2 + \gamma_2,$   
         $\alpha, (\beta - \beta_2), (\gamma - \gamma_2)$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \beta_2 - \delta(\beta) + \gamma_2,$   
         $-\beta_2, (\gamma - \gamma_2), -(\alpha - \delta(\alpha))$ )  
  
    **yield** Gilbert3D( $p + \alpha - \delta(\alpha) + \gamma_2 - \delta(\gamma),$   
         $-\gamma_2, -(\alpha - \alpha_2), \beta_2$ )  
  
**end function**

---

---

**Algorithm 8** 3D Generalized Hilbert Function (Gilbert3D)

---

```
#  $p, \alpha, \beta, \gamma \in \mathbb{Z}^3$ 
function GILBERT3D( $p, \alpha, \beta, \gamma$ )

    # Parity of dimensions
     $\alpha_0 \leftarrow (|\alpha| \bmod 2)$ 
     $\beta_0 \leftarrow (|\beta| \bmod 2)$ 
     $\gamma_0 \leftarrow (|\gamma| \bmod 2)$ 

    # Base cases
    if  $((|\alpha| \equiv 2) \text{ and } (|\beta| \equiv 2) \text{ and } (|\gamma| \equiv 2))$ 
        return Hilbert3D( $p, \alpha, \beta, \gamma$ )
    return Gilbert2D( $p, \beta, \gamma$ ) if  $(|\alpha| \equiv 1)$ 
    return Gilbert2D( $p, \alpha, \gamma$ ) if  $(|\beta| \equiv 1)$ 
    return Gilbert2D( $p, \alpha, \beta$ ) if  $(|\gamma| \equiv 1)$ 

    # Eccentric cases
    if  $(3|\alpha| > 5|\beta|) \text{ and } (3|\alpha| > 5|\gamma|)$ 
        return  $S_0(p, \alpha, \beta, \gamma)$ 
    if  $(2|\beta| > 3|\gamma|) \text{ or } (2|\beta| > 3|\alpha|)$ 
        return  $S_2(p, \alpha, \beta, \gamma)$ 
    if  $(2|\gamma| > 3|\beta|)$ 
        return  $S_1(p, \alpha, \beta, \gamma)$ 

    # Bulk recursion
    return  $J_0(p, \alpha, \beta, \gamma)$  if  $(\gamma_0 \equiv 0)$ 
    return  $J_1(p, \alpha, \beta, \gamma)$  if  $(\alpha_0 \equiv 0) \text{ or } (\beta_0 \equiv 0)$ 
    return  $J_2(p, \alpha, \beta, \gamma)$ 
end function
```

---