Reformulating Multiplication for Mandelbrot-like Sets in 3D

Eric Zimmermann[1] and Stefan Bruckner[2]

Institute for Visual & Analytic Computing, University of Rostock, Germany
[1]e.zimmermann@uni-rostock.de, [2]stefan.bruckner@uni-rostock.de

Abstract

The Mandelbrot set, named after Benoît Mandelbrot, led to a broad range of research as well as inspiring and appealing results. Even though there is no analogue description in 3-space, several extensions were considered. In this article, we recall the spherical representation and discuss a possible reformulation of trigonometric functions to improve the rendering performance. Afterwards, we set up a multiplication in 3-space, motivated by the cross product and its relation to differential forms. These provide the possibility to choose two functions as coefficients to achieve a varying multiplication in space, which we experimentally investigate using two iteration schemes and several function pairs. Finally, we turn to Julia sets, which can be achieved due to a minor change in the usual Mandelbrot iteration scheme and we explore examples w.r.t. function and constant choices.

Spherical Representation in 3D

The Mandelbrot set (see Figure 1a), introduced by Benoît Mandelbrot, has captivated mathematicians and inspired computer artists since the 1980s, providing a visually intriguing foundation for generative art. The accessibility of computational tools allowed artists to explore and visualize complex fractal structures, significantly enriching the field of digital aesthetics and computer-generated art [7]. This paper builds on these foundations with the aim of expanding the design space in order to offer artists and researchers enhanced possibilities for creative experimentation.

The Mandelbrot set consists of all the points c in the complex plane for which the scheme $z_{n+1} \to z_n^2 + c$ with $n \in \mathbb{N}_0$ and starting value $z_0 = 0$ is bounded. Various approaches to explore analogues and variations in 3D led to very interesting and visually appealing results like the Mandelbulb or projections of Quaternion Julia Sets into 3D [2].

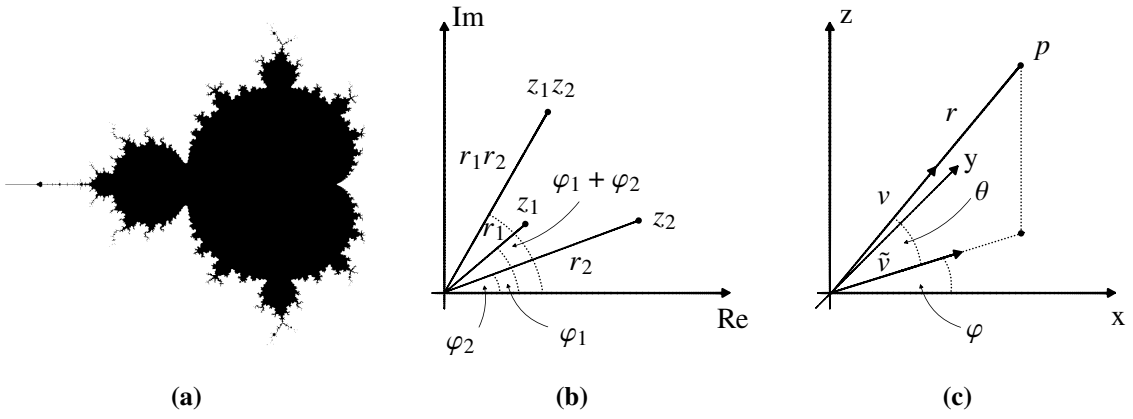
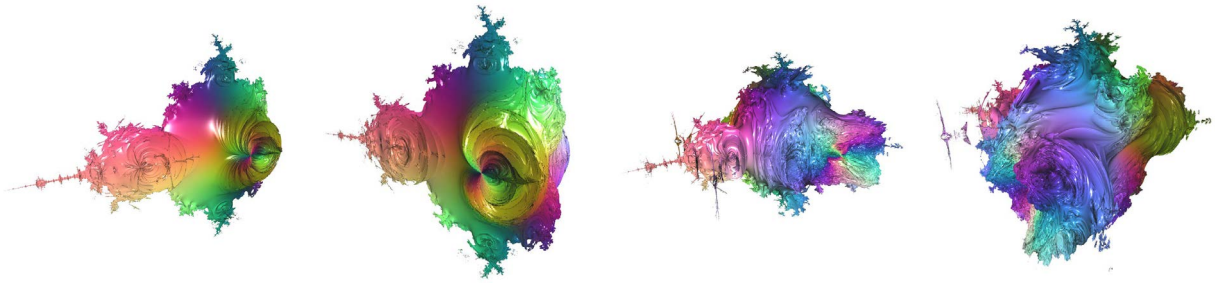


Figure 1: *(a) shows a rendering of the Mandelbrot set in the complex plane after 20 iterations, (b) an illustration of the rotation and stretching performed during complex multiplication, and (c) entities used in the spherical representation in 3D.*



(a) **(b)** **(c)** **(d)**

Figure 2: *Renderings of two versions of the multiplication using spherical coordinates in 3D from two viewpoints each. The second version carries* $-\sin(\theta_1 + \theta_2)$ *in the third coordinate in Equation (2). Note that the viewpoints are similar for (a) and (c) as well as (b) and (d).*

One such extension to 3D was proposed by Daniel White in 2007 [2] using spherical coordinates to imitate the behavior of the multiplication in the complex plane, which can be given for two numbers $z_1, z_2 \in \mathbb{C}$ in the complex plane $\mathbb{C}$ as

$$z_1 z_2 = (x_1 + iy_1)(x_2 + iy_2) = x_1 x_2 - y_1 y_2 + i(x_1 y_2 + x_2 y_1)$$

representing $z_j = x_j + iy_j$, $j = 1, 2$, with real and imaginary parts x_j and y_j, respectively, or

$$z_1 z_2 = r_1 \exp(i\varphi_1) r_2 \exp(i\varphi_2) = r_1 r_2 \exp(i(\varphi_1 + \varphi_2))$$

in polar form using $z_j = r_j \exp(i\varphi_j)$, $j = 1, 2$, with absolute values r_j (individually also known as modulus or magnitude) and arguments φ_j. The polar form provides a way to interpret the multiplication as a rotation (adding the arguments) and stretching (multiplying the absolute values) in the complex plane, see Figure 1b. The extension in 3D makes use of two (angular) arguments φ and θ to reflect the rotational part. For a point $p = (x, y, z) \in \mathbb{R}^3$ we can obtain the absolute value $r \in \mathbb{R}_{\geq 0}$ and arguments $\varphi \in [-\pi, \pi]$ and $\theta \in [-\pi/2, \pi/2]$ by

$$r = \sqrt{x^2 + y^2 + z^2}, \quad \varphi = \operatorname{arctan2}(y, x), \quad \text{and} \quad \theta = \arcsin(z/r), \tag{1}$$

where arctan2 gives the angular argument φ in the xy-plane between $-\pi$ and π except for the undefined case in which $x = y = 0$ and arcsin provides the opening angle θ above the xy-plane when $r > 0$, see Figure 1c. The multiplication of two points $p_1, p_2 \in \mathbb{R}^3$ can then be written as

$$p_1 p_2 = r_1 r_2 \begin{pmatrix} \cos(\varphi_1 + \varphi_2)\cos(\theta_1 + \theta_2) \\ \sin(\varphi_1 + \varphi_2)\cos(\theta_1 + \theta_2) \\ \sin(\theta_1 + \theta_2) \end{pmatrix}. \tag{2}$$

This representation corresponds to the one used by Barrallo [2], whereas another common representation interchanges $\cos(\theta_1 + \theta_2)$ with $\sin(\theta_1 + \theta_2)$ and vice versa and is therefore another parametrization caused by a parameter shift.

As the complex plane and $\mathbb{R}^3$ have componentwise addition, we get the iteration scheme

$$p_{n+1} = p_n^2 + c$$

with starting value $p_0 = (0, 0, 0)$, $c \in \mathbb{R}^3$, and $n \in \mathbb{N}_0$. The divergence condition, in which point c does not belong to the set if for some n the absolute value of p_n is greater or equal to 2, applies as well [2]. Figure 2 shows renderings also discussed by Barrallo [2].

Rendering in 3D

The included 3D renderings were generated with GPU-based raymarching implemented using the web-based 3D eninge *babylon.js*. The full source code is avaiable on GitHub [10]. We use an initial step size $\varepsilon \in \mathbb{R}_{\geq 0}$ and employ the clarity function (Equation (16)) in Hart et al. [6] to adjust the sampling resolution using the parameters $\alpha \in \mathbb{R}_{\geq 0}$ and $\delta \in \{0, 1, 2\}$. Note that we do not use the distance estimator from Equation (8) in Hart et al. [6], also used by Crane [5], because it is not clear whether such estimator is applicable in our scenario involving the upcoming multiplication changes and the variants of iteration schemes. Instead we use constant step sized raymarching with initial value ε and we allow step size changes caused by the clarity function which scales the Euclidean distance from the eye to the current location on the ray [6]. Each ray is intersected with a sphere of radius $\mathfrak{r} \in \mathbb{R}_{\geq 0}$, the escape radius, and this radius is also used for the divergence condition. Further, we declare the maximal number of iterations $\mathfrak{i} \in \mathbb{N}_0$. The default values are $\varepsilon = 5.e-5$, $\alpha = \varepsilon$, $\delta = 1$, $\mathfrak{i} = 10$, and $\mathfrak{r} = 2$ and deviations from those are mentioned accordingly. In particular, varying the escape radius enables the rendering of larger objects whose geometry is not strictly tied to the conventional value of 2, especially in the context of the modified iteration and multiplication schemes introduced later. For the shading we make use of the normal estimation by central differences also used by Crane [5], in which we find 6 points around the point in question, which we process in the same respective scheme with 10 iterations each to obtain an approximated normal. The distance to the 6 neighboring points is taken w.r.t. the distance obtained by the clarity function [6]. For the color at a point p, we normalize the vector $f(p) \times g(p)$ (explained in the upcoming reformulation of the multiplication), modify it s.t. its entries lie in $[0, 1]$, and use it to assign RGB values. In this way, a color indicates in which direction the vector shows and repeating color patterns around a point, for instance in Figure 2b, could reflect properties of multiplicities around poles or zeros encoded in domain coloring [8]. The final color values are then determined by the Phong reflection model.

Reformulations of the involved computations—like using polynomials [1, 2, 9]—can improve the rendering performance. To circumvent the evaluation of trigonometric functions in Equation (2) and the calculation of inverses from Equation (1), our rephrasing makes use of the definition of the angle α between vectors v_1, v_2 defined as $\cos(\alpha) =< v_1, v_2 > /(\|v_1\| \, \|v_2\|)$ (using the scalar product $< \cdot, \cdot >$ and the Euclidean norm $\|\cdot\|$) and the addition theorems for the cosine and sine equations. Suppose we are given the two points $p_1, p_2 \in \mathbb{R}^3$ with absolute values $r_j = \left\|p_j\right\|$, normalized representatives $v_j = p_j/r_j = (v_j^x, v_j^y, v_j^z)^T$ and normalized projections $\tilde{v}_j = (v_j^x, v_j^y, 0)^T / \left\|(v_j^x, v_j^y, 0)\right\|$ for $j = 1, 2$, see also Figure 1c. Note that here we used the superscript to identify the coordinates. Now we are able to write the trigonometric terms in Equation (2) as

$$
\begin{aligned}
\cos(\varphi_1 + \varphi_2) &= \cos(\varphi_1)\cos(\varphi_2) - \sin(\varphi_1)\sin(\varphi_2) \\
&=< \tilde{v}_1, v_x >< \tilde{v}_2, v_x > - < \tilde{v}_1, v_y >< \tilde{v}_2, v_y > \\
&= \tilde{v}_1^x \tilde{v}_2^x - \tilde{v}_1^y \tilde{v}_2^y, \\
\cos(\theta_1 + \theta_2) &= \left(v_1^x \tilde{v}_1^x + v_1^y \tilde{v}_1^y\right)\left(v_2^x \tilde{v}_2^x + v_2^y \tilde{v}_2^y\right) - v_1^z v_2^z, \\
\sin(\varphi_1 + \varphi_2) &= \tilde{v}_1^y \tilde{v}_2^x + \tilde{v}_2^y \tilde{v}_1^x, \quad \text{and} \\
\sin(\theta_1 + \theta_2) &= v_1^z\left(v_2^x \tilde{v}_2^x + v_2^y \tilde{v}_2^y\right) + v_2^z\left(v_1^x \tilde{v}_1^x + v_1^y \tilde{v}_1^y\right),
\end{aligned}
$$

with $v_x = (1, 0, 0)^T$ and $v_y = (0, 1, 0)^T$ and a more detailed description for the first one. This also provides the option to extract individual terms like $\cos(\varphi_1)$. Note that the behavior of the undefined cases in Equation (1) is reflected in v_j and $\tilde{v}_j$. In general, we would assume that angular values do not change during multiplication if they are undefined anyway. Thus, setting for a point $p = (x, y, z)$ the value $\varphi = 0$ when $x = y = 0$ and $\theta = 0$ when $x = y = z = 0$ won't change the respective angular components during multiplication. In terms of the expressions above we declare $\tilde{v}_j = (1, 0, 0)$ for $p_j = (0, 0, z_j)$ and $v_j = (1, 0, 0)$, $\tilde{v}_j = v_j$ for $p_j = (0, 0, 0)$. When we then multiply p with itself, we obtain $\cos(2\varphi) = 1$, $\sin(2\varphi) = 0$, $\sin(2\theta) = 0$ for both cases and $\cos(2\theta) = -1$ and $\cos(2\theta) = 1$ for $p = (0, 0, z)$ and $p = (0, 0, 0)$, respectively.

Rephrasing Multiplication

The basic idea of using spherical coordinates to represent the multiplication is to translate the behavior of complex multiplication to 3D involving rotations around two angles. However, the product in Equation (2) lacks properties to account for a valid extension into 3D, cf. the undefined cases in Equation (1). A possible product to consider is the geometric product used in geometric algebra working with multivectors. The latter include bivectors which are obtained via the exterior product (also called wedge product) and thus relate to the following considerations using differential forms with smooth coefficients.

While keeping the basic idea of iteration, we would like to alter the product. A candidate in 3D is the cross product, which is defined for vectors $u, v \in \mathbb{R}^3$ as $u \times v = (u_2v_3 - u_3v_2, u_3v_1 - u_1v_3, u_1v_2 - u_2v_1)$. This is not suitable, because multiplying a vector with itself yields the 0-vector. To make use of the cross product, we modify the coordinates of the points before multiplying them. Here we restrict ourselves to a single point, as used in the original iteration scheme and it aligns with the idea that the cross product is a special case of the wedge product applied to differential forms evaluated at a certain point [4]. A differential k-form, $k \in \mathbb{N}_0$, is a multilinear alternating function operating on tangent vectors and returning a number. For instance $\omega = f_1(x, y, z)dx + f_2(x, y, z)dy + f_3(x, y, z)dz$ is a 1-form in $\mathbb{R}^3$ including differentials dx, dy, dz and smooth scalar functions f_j for $j = 1, 2, 3$. The multiplication of a k_1-form and k_2-form producing a $(k_1 + k_2)$-form is carried out by the wedge product. Assigning constant values for the coefficients in ω and ν produces coefficients in the 2-form $\omega \wedge \nu$ which coincide with the components of the cross product of vectors consisting of the same coefficients used for ω and ν, as in $\mathbb{R}^3$ 1- and 2-forms are associated with vectors [4].

For our coefficients we use two functions $f, g : \mathbb{R}^3 \to \mathbb{R}^3$, not necessarily smooth, which we evaluate at a given point p, i.e. $f(p) = (f_1(p), f_2(p), f_3(p))$ and g likewise, and multiply via the cross product $f \times g$. With componentwise addition, we set up two iteration schemes to decide whether a point $c \in \mathbb{R}^3$ belongs to our object using starting value p_0 and functions f, g. The first one is

$$\begin{aligned} p_0 &= f(c) \times g(c) \\ p_{n+1} &= p_n \times g(c) \quad \forall n \in \mathbb{N} \end{aligned} \tag{3}$$

making use of the initial idea of the cross product taken between p_n and c processed via f and g beforehand. The second scheme is

$$\begin{aligned} p_0 &= c \\ p_{n+1} &= f(p_n) \times g(p_n) + c \quad \forall n \in \mathbb{N} \end{aligned} \tag{4}$$

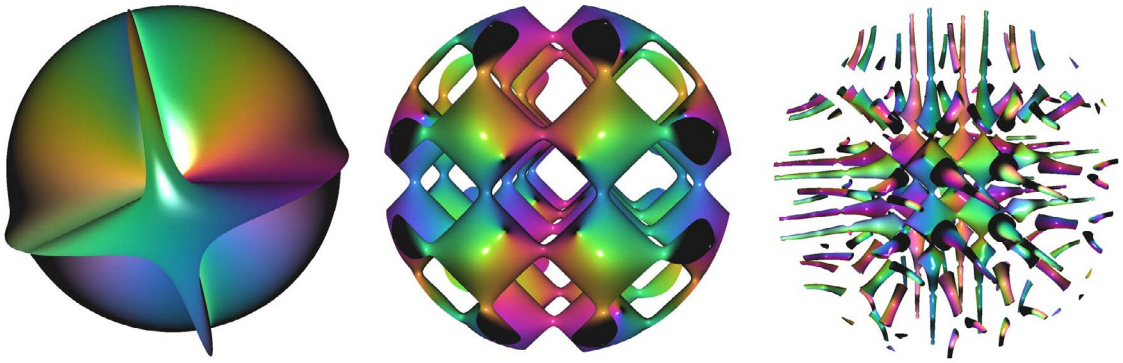
which reflects the original iteration process of the Mandelbrot set, yet starting at the point $c \in \mathbb{R}^3$, which belongs to the sequence of the original Mandelbrot set anyway. Therefore, in our scenario the evolution is caused because we let the multiplication change throughout space using differential forms and their coefficients f and g, instead of letting the cross product act globally.

Reformulating Old and Finding New Expressions

At this point we have two iteration schemes and the freedom to choose two functions f and g, for which we omit writing down their arguments, i.e., write f instead of $f(x, y, z)$, and we label different examples with letter subscripts, for instance f_A and g_A represent one example. For the upcoming values we obtain for a point c its magnitude r and angular arguments φ and θ as stated in Equation (1).

At first, we express the spherical representation given in Equation (2). This can be done by using the scheme explained in Equation (4) and setting

$$f_A = r^2 \begin{pmatrix} -\sin(2\varphi) \\ \cos(2\varphi) \\ 0 \end{pmatrix} \quad \text{and} \quad g_A = \begin{pmatrix} \frac{-\sin(2\theta)}{\cos(2\varphi)} \\ 0 \\ \cos(2\theta) \end{pmatrix}.$$



(a) **(b)** **(c)**

Figure 3: *Renderings obtained using the scheme from Equation (3) with (a) $f_A, g_A, \mathfrak{i} = 100$, and $\mathfrak{r} = 10$, (b) $f_B, g_B, \mathfrak{i} = 100$, and $\mathfrak{r} = 6$, and (c) $f_C, g_C, \mathfrak{i} = 100$, and $\mathfrak{r} = 10$.*

Evaluating the cross product from Equation (4) with f_A and g_A reduces to the description in Equation (2), cf. Figure 2, as may be verified explicitly by expanding the cross product as

$$f_A \times g_A = r^2 \begin{pmatrix} \cos(2\varphi)\cdot\cos(2\theta) - 0\cdot 0 \\ 0\cdot\left(-\frac{\sin(2\theta)}{\cos(2\varphi)}\right) - (-\sin(2\varphi))\cdot\cos(2\theta) \\ -\sin(2\varphi)\cdot 0 - \cos(2\varphi)\cdot\left(-\frac{\sin(2\theta)}{\cos(2\varphi)}\right) \end{pmatrix} = r^2 \begin{pmatrix} \cos(2\varphi)\cos(2\theta) \\ \sin(2\varphi)\cos(2\theta) \\ \sin(2\theta) \end{pmatrix}$$

evaluated at one point. Note that g_A is not defined for $\varphi \in \left\{\pm\frac{\pi}{4}, \pm\frac{3\pi}{4}\right\}$ and we declare a point with such value for φ failing the divergence condition and consequently it does not belong to the set. In contrast, the functions f_A and g_A processed with the scheme from Equation (3) cause an appearance which is very different from the one obtained using the scheme from Equation (4) extending in each direction when increasing $\mathfrak{r}$, cf. Figure 3a. An extension through space can also be observed for the following two pairs of functions

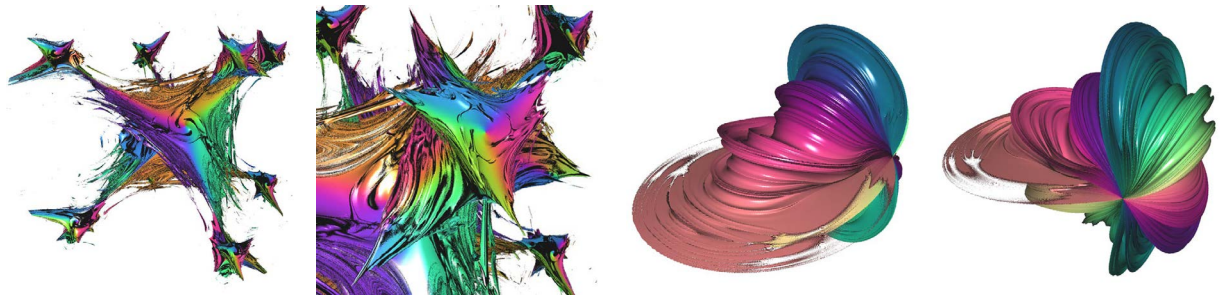
$$f_B = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad g_B = \begin{pmatrix} \cos(x) \\ \cos(y) \\ \cos(z) \end{pmatrix} \quad \text{and} \quad f_C = \begin{pmatrix} x^2 \\ y^2 \\ z^2 \end{pmatrix}, \quad g_C = g_B,$$

evaluated in the same scheme, see Figures 3b and 3c, respectively. These objects converge rather fast and increasing the sphere for rendering in $\mathfrak{r}$ suggests that especially the second object extends through $\mathbb{R}^3$.

To explore additional examples for the scheme in Equation (4), we start with functions f_C and g_C. The resulting object, see Figure 4, includes some kind of swirl which can also be seen from the opposite side but slightly rotated. However, not all function pairs elaborated with the scheme from Equation (3) have visually appealing counterparts when processed with the scheme from Equation (4). Returning to the question of an extension of the Mandelbrot set to 3-space, we could think of a pair of functions forcing the Mandelbrot set in the xy-plane, i.e.

$$f_D = r^2 \begin{pmatrix} -\sin(2\varphi) \\ \cos(2\varphi) \\ 0 \end{pmatrix} \quad \text{and} \quad g_D = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

In Figures 4c and 4d, the overall shape of the Mandelbrot set appears in the xy-plane, yet it also exhibits arcs extending in the z-direction. These arcs are reminiscent of the 4D quaternionic example's projection into 3D, which appears as a rotation of the Mandelbrot set around the x-axis [2].



(a) **(b)** **(c)** **(d)**

Figure 4: *Renderings obtained using the scheme from Equation (4) with (a), (b)* f_C, g_C*,* $\mathfrak{i} = 100$*, and* $\mathfrak{r} = 5$ *and (c), (d)* f_D, g_D*, and* $\mathfrak{r} = 8$*.*

Turning to Julia Sets

The iteration process used for the Mandelbrot set can be modified slightly by iterating the points in the complex plane while fixing an added constant c to explore, for instance, the dynamics of quadratic functions $z \mapsto z^2 + c$ [6]. In this case, we are looking for the points z not traveling to infinity during the process w.r.t. a fixed constant c. The resulting fractals are called Julia sets. With a minor change, Equation (4) becomes

$$\begin{aligned} p_0 &= c \\ p_{n+1} &= f(p_n) \times g(p_n) + \mathfrak{c} \quad \forall n \in \mathbb{N} \end{aligned} \tag{5}$$

for every point $c \in \mathbb{R}^3$ and a fixed constant $\mathfrak{c} \in \mathbb{R}^3$. Note that for simplicity we treated c as the iterated point rather than as the constant. Hence it is only a minor change to the overall procedure.

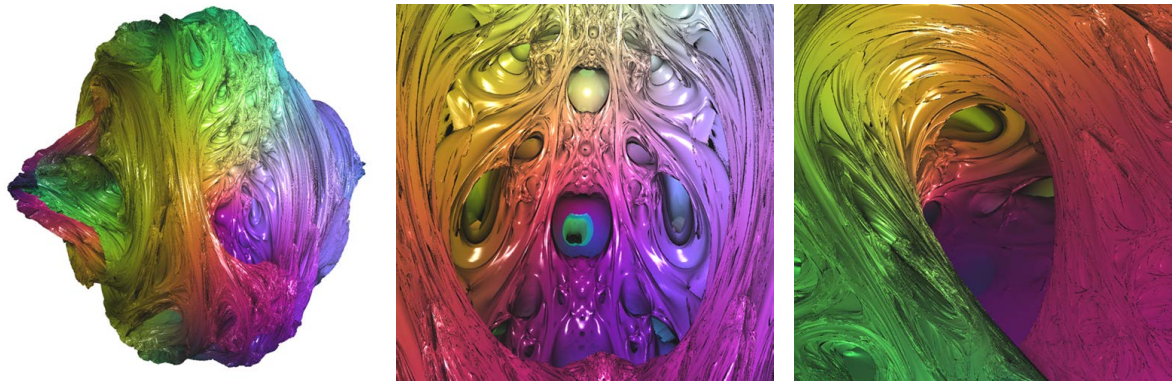
As the Julia sets are defined in the complex numbers we experiment with known constants for Julia sets of the quadratic functions. We place these constants in the x and y coordinate of $\mathfrak{c}$ in the same way the traditional Mandelbrot set is reflected in the xy-plane expressed by the spherical representation, cf. Equation (2).

A first example occurs using constant $\mathfrak{c} = (-1, 0, 0)$ shown in Figure 5, where the related point $(-1, 0)$ is the center of a 2-dimensional ball in the Mandelbrot set, cf. Figure 1a. Here we can find an interesting structural behavior. The pattern of the corresponding Julia set in 2D is reflected in the balls in 3D, decreasing in diameter attached to each other along a line which is the x-axis. Along the z-axis, we can identify flattened repeating pieces, we may call them flats. The balls and flats, however, repeat not just along the axes. It seems that the balls with attached flats occur as smaller copies across the larger balls and we can find smaller flats distributed on the larger ones, giving the whole object the self-similarity fractals are known for. Variations of the constants $\mathfrak{c}$ allow a broad range of objects even yielding some which appear more dense and penetrated by tunnels. An example can be seen in Figure 6 where the indentation, occurring on opposite sides, is littered by entrances to tunnels inviting the viewer to navigate through them.

The examples we have considered so far (using f_A and g_A) are not dependent on the reformulation of the multiplication, i.e., we can render the same objects using the spherical representation (Equation (2)). However, varying f and g for Julia sets also allows us to generate very different objects like the ones displayed in Figure 7. Here the objects in Figures 7a and 7b remind us on their counterparts in Figures 3c and 4, yet both partly reflected in the different schemes given in Equations (3) and (4). The third one in Figure 7c shows a very different appearance and hints the possible variety of objects we could achieve with f and g. The last example (Figure 7d) illustrates an alteration in the third coordinate of $\mathfrak{c}$ where increasing the value 0.31 thins out the rings further until the object disappears at the latest at 0.348. Hence, the scheme used to describe Julia sets and the choices for f, g, and $\mathfrak{c}$ extend the space for exploration even further.

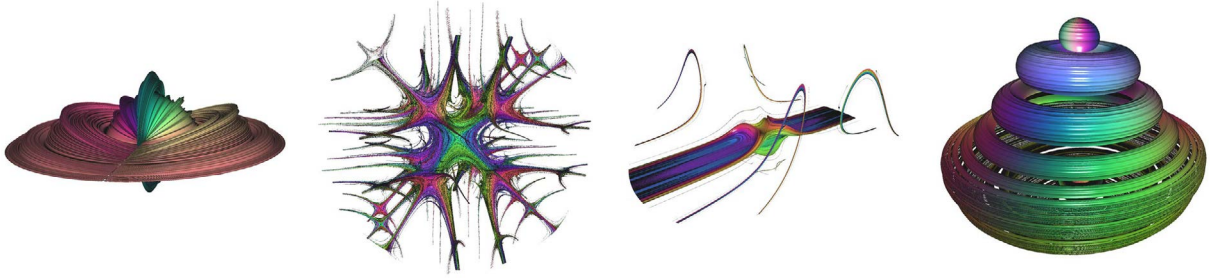
(a) **(b)** **(c)**

Figure 5: *Renderings obtained using the scheme from Equation (5) with (a) f_A, g_A, $\mathfrak{i} = 100$ and $\mathfrak{c} = (-1, 0, 0)$. The encircled labeled regions in (a) are magnified in (b) and (c). Further renderings can be found in the supplementary material.*



(a) **(b)** **(c)**

Figure 6: *Renderings obtained using the scheme from Equation (5) with (a) f_A, g_A and $\mathfrak{c} = (0.39, 0.2, 0)$. The object appears more solid possibly penetrated by tunnels with entrances on the surface (b), (c). Note that for visibility the iteration numbers $\mathfrak{i}$ are chosen 10, 50, and 15, respectively.*



(a) **(b)** **(c)** **(d)**

Figure 7: *Renderings obtained using the scheme from Equation (5) with (a) f_D, g_D, (b) f_C, g_C, and $\mathfrak{r} = 10$, and (c) $f_E = (x^2y, y^2, z^2)^T$, $g_E = (1, \cos(z), 1)^T$, and $\mathfrak{r} = 10$. All three used $\mathfrak{c} = (-1, 0, 0)$. (d) shows a result of f_A and g_A with $\mathfrak{c} = (0, 0, 0.31)$. All examples used $\mathfrak{i} = 100$.*

Summary and Conclusions

Motivated by the Mandelbrot set's spherical extension into 3D, we recalled its description and discussed a possible way to rewrite the trigonometric functions to improve performance. Due to the missing correspondence of a 3-dimensional product, we used the relation of the cross product and differential 1-forms to describe two iteration approaches and the freedom to set functions f and g, to, on the one hand, reformulate the spherical representation and on the other hand allow new ones. A minor change in the iteration process let us turn to Julia sets, where we considered different constants and their visual appearances.

The flexibility to select and combine different functions f and g within distinct iterative schemes enables the generation and exploration of novel geometric structures, potentially appealing to artists. Nonetheless, understanding the intricate relationship between the chosen functions and their resulting geometric forms remains challenging in both directions: determining how specific functions affect the resulting shapes and identifying suitable functions to generate desired properties. Exploring this relation between functions and their results by sampling and clustering the space of functions [3] is an interesting direction for future research.

Geometric objects, particularly detailed fractals whose complexity makes physical realization impractical, are well-suited for presentation within virtual environments and digital art exhibitions. Such settings enable audiences to engage interactively with intricate structures otherwise restricted to static two-dimensional renderings. Furthermore, as the rendering of fractals is still computationally demanding—even on current graphics hardware—continued advancements in computational methods, such as improved validation of distance estimates [6] [5], would be beneficial.

Moreover, while we believe the color coding used throughout the paper represents a meaningful choice, it is just one of many possibilities and a more thorough investigation of this aspect could be an interesting topic for further investigation as well.

References

[1] R. Alonso-Sanz. "A Glimpse of the Mandelbulb with Memory." *Complex Systems*, vol. 25, no. 2, 2016.

[2] J. Barrallo. "Expanding the Mandelbrot Set into Higher Dimensions." *Bridges Conference Proceedings*. Pécs, Hungary, July 24–28 2010. pp. 247–254. http://archive.bridgesmathart.org/2010/bridges2010-247.html.

[3] S. Bruckner and T. Möller. "Result-Driven Exploration of Simulation Parameter Spaces for Visual Effects Design." *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, 2010, pp. 1468–1476.

[4] P. Collier. *A Beginner's Guide to Differential Forms*, 1st ed. Incomprehensible Books, 2021.

[5] K. Crane. "Ray Tracing Quaternion Julia Sets on the GPU." 2005. https://www.cs.cmu.edu/~kmcrane/Projects/QuaternionJulia/.

[6] J. C. Hart, D. K. Sandin, and L. H. Kauffman. "Ray tracing deterministic 3-D fractals." *Computer Graphics (SIGGRAPH '89 Proceedings)*, vol. 23, no. 3, 1989, pp. 289–296.

[7] C. A. Pickover. *Computers, Pattern, Chaos, and Beauty: Graphics from an Unseen World*. Dover Publications, 2001.

[8] K. Poelke and K. Polthier. "Domain Coloring of Complex Functions: An Implementation-Oriented Introduction." *IEEE Computer Graphics and Applications*, vol. 32, no. 5, 2012, pp. 90–97.

[9] I. Quilez. "Mandelbulb." 2009. https://iquilezles.org/articles/mandelbulb/.

[10] E. Zimmermann and S. Bruckner. "Vractal-explorer." 2025. https://github.com/e-zimmermann/vractal-explorer.