Sistemas dinámicos discretos e iteradas de funciones complejas: fractales, caos y estabilidad.

David Armenteros Soto
Tutor: Pedro José Torres Villarroya

[1]Universidad de Granada
Facultad de ciencias

[2]Universidad de Granada
Escuela Técnica Superior de Ingenierías Informática y Telecomunicación

1 de diciembre de 2021

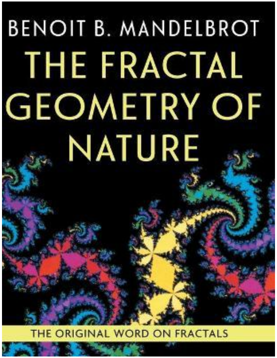
Índice general

UNIVERSIDAD DE GRANADA

Índice general

Introducción

En la naturaleza con frecuencia aparecen formas que somos incapaces de describir. Si observamos a nuestro alrededor, percibimos objetos que repiten un patrón indefinidamente.



Motivación

La geometría fractal cambiará a fondo su visión de las cosas. Seguir leyendo es peligroso. Se arriesga a perder definitivamente la imagen inofensiva que tienen las nubes, bosques, galaxias, hojas, plumas, flores, rocas, montañas y muchas otras cosas. Jamás volverá a recuperar las interpretaciones de todos estos objetos que hasta ahora le eran familiares.

Michael Barnsley

UNIVERSIDAD DE GRANADA

Objetivos

- Definir, explicar y exponer nociones básicas de los sistemas dinámicos discretos: iteradas, puntos fijos, puntos periódicos, estabilidad, ...
- Ejemplificar la teoría de sistemas con ejemplos lineales y no lineales mediante el uso de herramientas informáticas para la representación de mapas, iteradas, órbitas, puntos fijos y periódicos.
- Introducir la teoría del caos, representar diagramas de bifurcación y calcular el exponente de Lyapunov.
- Explicar y representar los conjuntos de Julia y de Mandelbrot y fractales de Newton.
- Generar y visualizar fractales de dos dimensiones mediante su construcción clásica y sistemas de Lindemayer.
- Formalizar la construcción de fractales mediante SFI.
- Definir y explicar la dimensión de Hausdorff. Implementar el algoritmo box counting para el cálculo de esta dimensión.
- Visualizar fractales tridimensionales (Mandelbulb, Mandelbox y cuaterniones de Julia) mediante ray tracing.

Índice general

UNIVERSIDAD DE GRANADA

Sistemas dinámicos discretos

Definición

Sea $X \subset \mathbb{R}^n$, $X = (X, d)$ un espacio métrico y $f : X \to X$ una función continua. Un **sistema dinámico discreto** es un par de la forma (X, f) donde f es el **mapa** asociado al sistema.

Conceptos previos: mapa, iterada, órbita, valor inicial, punto fijo, punto periódico, estabilidad, atracción, repulsión, hiperbolicidad, caos, ...

Figura: [Canva] Conceptos básicos

Índice general

UNIVERSIDAD
DE GRANADA

Sistemas dinámicos complejos

- Conjunto de Julia
- Conjunto de Mandelbrot

Figura: Gaston Julia

Figura: Benoit B. Mandelbrot

UNIVERSIDAD DE GRANADA

Conjunto de Julia

Definición

Sea $f : \mathbb{C} \to \mathbb{C}$ un mapa continuo. Se define el **conjunto de Julia relleno** asociado al mapa f:

$$K(f) = \left\{ z \in \mathbb{C} : O^{+}(z)\, acotada \right\}$$

Definición

Se define el **conjunto de Julia** $J(f)$ (asociado al mapa f) como la frontera del conjunto de Julia relleno $K(f)$. Equivalentemente, $J(f)$ se puede definir como la frontera del conjunto de escape:

$$E = \left\{ z \in \mathbb{C} : \lim_{n \to \infty} |f^{n}(z)| = \infty \right\}$$

El conjunto de Julia que analizaremos está ligado al mapa cuadrático:

$$z_{n+1} = f_c(z_n) = z_n^2 + c \quad con \quad c \in \mathbb{C}$$

UNIVERSIDAD DE GRANADA

Conjunto de Julia

Notación El conjunto de Julia relleno y el conjunto de Julia ligados al mapa $f_c(z)$ se denotan, respectivamente, como K_c y J_c.

Teorema

Dado $z \in \mathbb{C}$ y el mapa $f_c(z) = z^2 + c$. Si $|z| > \max\{2, |c|\}$ entonces:

$$\lim_{n\to\infty} |f_c^n(z)| = \infty$$

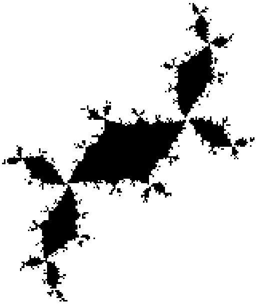
.

Sea $r(c) = \max\{|c|, 2\}$. Se conoce a $r(c)$ como el **radio umbral** de f_c.

UNIVERSIDAD DE GRANADA

Conjunto de Julia

1. Algoritmo para el conjunto de Julia relleno en blanco y negro
2. Algoritmo de iteración inversa (cerco del conjunto de Julia)



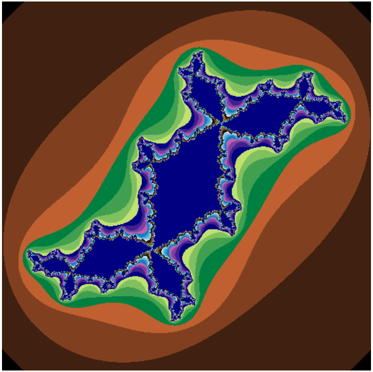


Figura: [Python] Conjunto de Julia relleno $K_{-0,1+0,8i}$ (**El conejo de Douday**)

Conjunto de Julia

Teorema

Las siguientes afirmaciones sostienen que:

1. Si $0 \in K_c$ entonces el conjunto de Julia J_c es **arcoconexo**
2. Si $0 \notin K_c$ entonces el conjunto de Julia J_c es **disconexo**

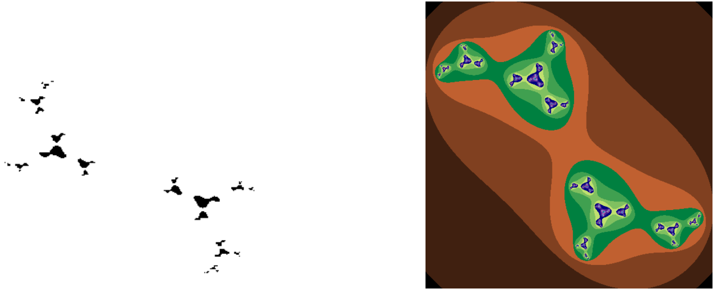


Figura: [Python] Conjunto de Julia relleno disconexo $K_{-1,2i}$

Conjunto de Mandelbrot

Definición

El **conjunto de Mandelbrot** consta de todos los valores de $c \in \mathbb{C}$ para los que la órbita de 0 bajo f_c está acotada, es decir, no escapa al infinito.

$$M = \left\{ c \in \mathbb{C} : O^+(0) \ \textit{acotada bajo}\ f_c \right\}$$

Corolario

Para el mapa cuadrático f_c, se tiene que:

1. $c \in M$ y el conjunto de Julia correspondiente J_c es **arcoconexo**.
2. $c \notin M$ y el conjunto de Julia correspondiente J_c es **disconexo**.

Definición

El corolario anterior proporciona una caracterización del conjunto de Mandelbrot:

$$M = \{ c \in \mathbb{C} : J_c \ \textit{es arcoconexo} \}$$

Conjunto de Mandelbrot

Teorema

Si f_c tiene un punto periódico atractivo entonces $c \in M$.

Teorema

El conjunto de Mandelbrot M está contenido en el disco cerrado de radio 2:

$$M \subset \{c \in \mathbb{C} : |c| \leq 2\}$$

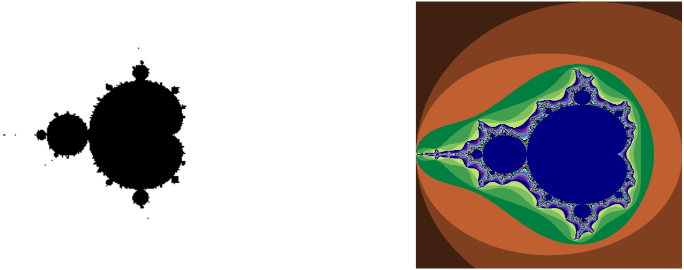


Figura: [Python] Conjunto de Mandelbrot

Índice general

UNIVERSIDAD DE GRANADA

Fractales

Definición

Un fractal es un objeto geométrico cuya estructura muestra **autosimilitud** (a veces llamada autosemejanza o autosimilaridad) y que se construye a partir de un patrón o imagen repetida a diferentes escalas.

Definición

Un fractal es un subconjunto de $\mathbb{R}^n$ cuya dimensión de Haussdorff es estrictamente mayor que la dimensión topológica.

- Sistema de funciones iteradas
- Sistemas de Lindemayer

Sistema de funciones iteradas

Definición

Sea $F_1, F_2, ..., F_N$ una familia de contracciones de $\mathbb{R}^2$ y $S \in H(\mathbb{R}^2)$ un subcojunto cerrado y acotado de $\mathbb{R}^2$. Se dice que el sistema $\{S, F\} = \left\{S, \cup_{i=1}^{N} F_i\right\}$ es un **sistema de funciones iteradas**.

Teorema

Si $F_1, F_2, ..., F_N$ son contracciones de $\mathbb{R}^2$, entonces exite un atractor global $A \in H(\mathbb{R}^2)$ para el mapa unión u operador de Hutchinson $F = \bigcup_{i=1}^{N} F_i$. Más concretamente, para todo $B \in H(\mathbb{R}^2)$, $F^n(B)$ converge a A con la distancia de Hausdorff.

UNIVERSIDAD DE GRANADA

Sistema de funciones iteradas

Algoritmos para la obtención del fractal asociado a un SFI

- SFI determinista: se basa en calcular directamente la sucesión $\{F^n(S)\}$
- SFI aleatorio: asignamos a cada contracción F_i una propabilidad $p_i > 0$ de ser seleccionado.

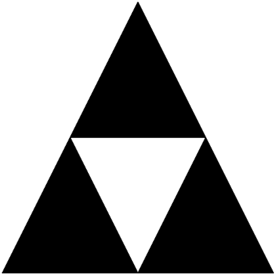


Figura: [Python] (i) Triángulo de Sierpiński (ii) Helecho de Barnsley

Sistema de Lindemayer

Definición

Se trata de una técnica para definir objetos complejos reemplazando sucesivamente partes de un objeto inicial usando reglas de escritura. Los sistemas de Lindenmayer están compuestos de tres componentes principales:

- Alfabeto
- Axioma
- Reglas de producción

UNIVERSIDAD DE GRANADA

Sistema de Lindemayer

Gráficos turtle

Instrucciones para la tortuga:

- **F**: avanza un paso (de cierta longitud l) dibujando el camino.
- **f**: lo mismo que F pero sin dibujar la trayectoria.
- **+**: gira a la izquierda (sentido antihorario) con un ángulo θ.
- -: gira a la derecha (sentido horario) con un ángulo θ.

Comando	Interpretación
L	$+F-F-F+$
R	$-F+F+F-$
S	$FF+F+FF-F-FF$
Z	$FF-F-FF+F+FF$
D	$--F++F$
E	$F--F++$

UNIVERSIDAD DE GRANADA

Sistema de Lindemayer

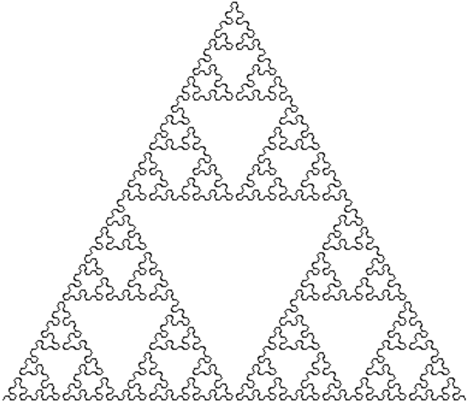


Figura: [Python] Punta de flecha de Sierpiński. Axioma: L , reglas de producción: $L \to +R - L - R+$, $R \to -L + R + L-$ y parámetros: $\theta = 60^o$

UNIVERSIDAD DE GRANADA

Sistema de Lindemayer

Estructuras ramificadas

Un punto de bifurcación comienza con el comando [y termina con].

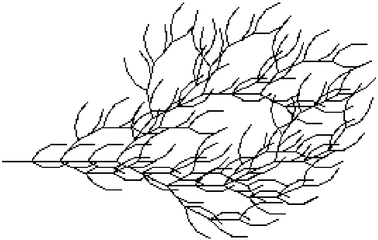


Figura: [Python] Ramificación de un arbusto. Axioma: F , reglas de producción: $F \to FF + [+F - F - F] - [-F + F + F]$ y parámetros: $\theta = 25^o$

UNIVERSIDAD DE GRANADA

Índice general

UNIVERSIDAD DE GRANADA

Dimensión fractal

Definición

Sea $A \in H(X)$ donde (X, d) es un espacio métrico. Para todo $\epsilon > 0$, sea $N(A, \epsilon)$ el menor número de bolas cerradas de radio $\epsilon > 0$ necesarias para cubrir A. Si

$$D = \lim_{\epsilon \to 0} \left\{ \frac{\ln(N(A,\epsilon))}{\ln(\frac{1}{\epsilon})} \right\}$$

existe entonces D es la dimensión fractal de A.

Dimensión Box Counting

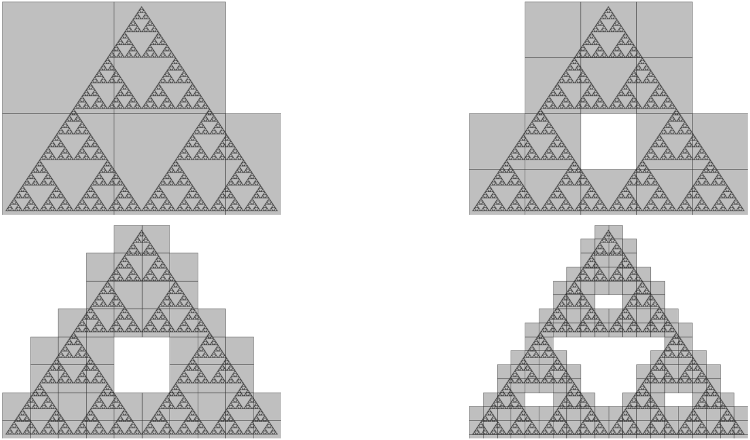


Figura: [Python] Algoritmo box counting para el triángulo de Sierpiński con cajas de dimensiones (i) 2^9 (ii) 2^8 (iii) 2^7 (iv) 2^6 píxeles.

Dimensión Box Counting

Dimensión	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2
Número	5	15	48	160	470	1352	3991	10993

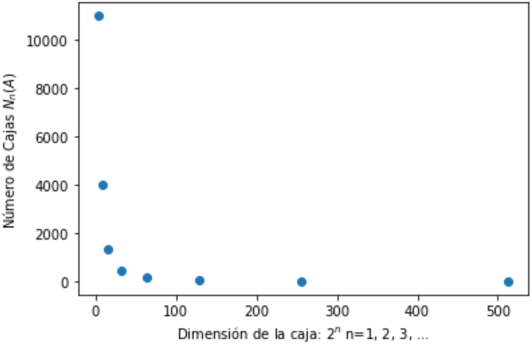


Figura: [Python] Conteo de cajas para el triángulo de Sierpiński

Dimensión Box Counting

La pendiente de la recta de mejor ajuste coincide con la dimensión box counting y su valor es 1,59

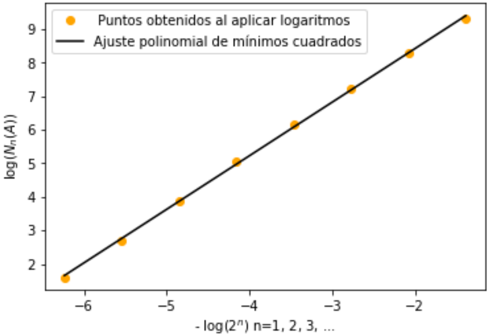


Figura: [Python] Recta de mejor ajuste de la gráfica logarítmica para el triángulo de Sierpiński

Índice general

Visualización de fractales con Ray Tracing

El **Ray Tracing** (o trazado de rayos) es una técnica de renderizado que puede simular de manera realista la iluminación de una escena y sus objetos al generar reflejos, refracciones, sombras e iluminación indirecta.

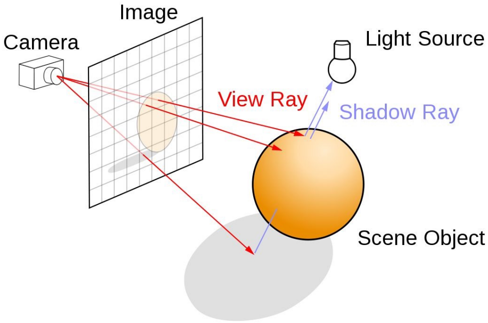


Figura: Conceptos básicos del trazado de rayos

Imagen

```
P3
3 2
255
255   0   0     0 255   0     0   0 255
255 255   0   255 255 255     0   0   0
```

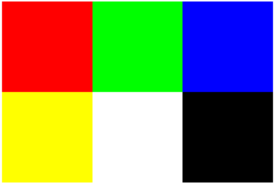


Figura: Formato de Imagen PPM

Rayos

Todo rayo tiene una posición de **origen** O, una posición de **destino** D y un vector **dirección** d que dice hacia dónde apunta. De esta forma, definimos un rayo como la recta:

$$r(t) = O + \frac{D - O}{\|D - O\|} t = O + dt \qquad O, D, d \in \mathbb{R}^3,\ t \in \mathbb{R}$$

Visualización de fractales con Ray Tracing

Modelo de reflexión de Phong

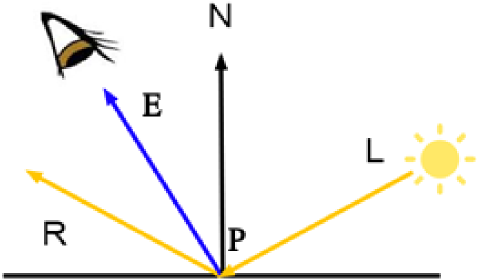


Figura: Vectores que intervienen en la iluminación de la escena

Visualización de fractales con Ray Tracing

Modelo de reflexión de Phong

- Ambiental
- Difusa
- Especular

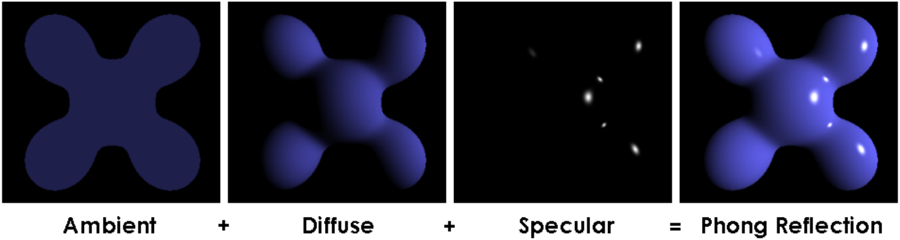


Figura: Iluminación de Phong

UNIVERSIDAD DE GRANADA

Visualización de fractales con Ray Tracing

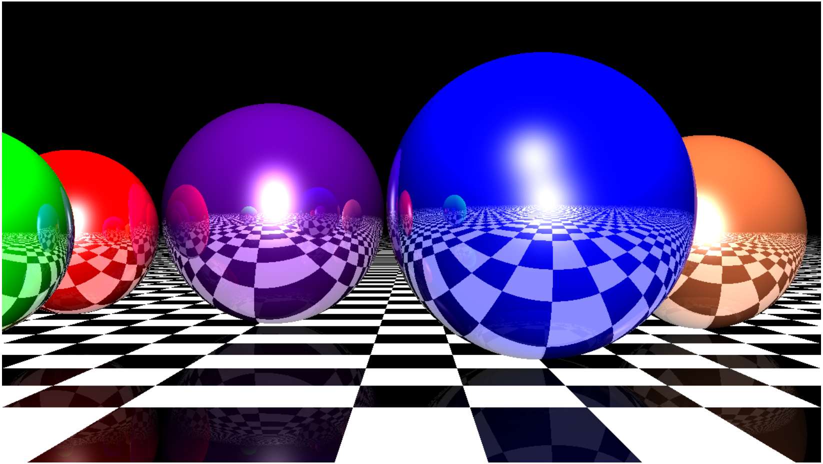


Figura: [Python] Esferas sobre un tablero de ajedrez.

Ray marching

El **ray marching** (o marcha de rayos) es una técnica con ciertas similitudes al ray tracing que renderiza objetos a través de campos de distancia.

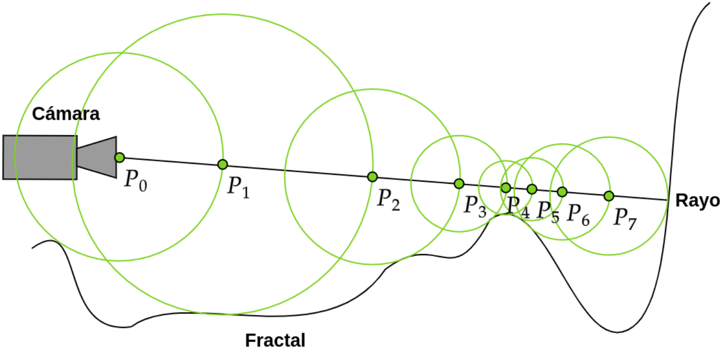


Figura: [Mathcha] Marcha de rayos

Ray marching

Aproximación de la normal de una superficie fractal

Un método común para aproximar la normal es calcular el gradiente de un punto en la superficie. Se calcula de la siguiente forma:

$$N_x = D_{x+\epsilon,y,z} - D_{x-\epsilon,y,z}$$

$$N_y = D_{x,y+\epsilon,z} - D_{x,y-\epsilon,z}$$

$$N_z = D_{x,y,z+\epsilon} - D_{x,y,z-\epsilon}$$

donde $\epsilon > 0$ y $D_{x,y,z}$ es el valor del estimador de la distancia en el punto (x, y, z). Dando como resultado el vector normal $N = (N_x, N_y, N_z)$.

UNIVERSIDAD DE GRANADA

Ray marching

Mandelbulb

El mandelbulb es una representación tridimensional del mapa $f_c(z) = z^n + c$. La potencia n-ésima del punto $p = (x, y, z)$ es

$$p^n = r^n(\sin(n\theta)\cos(n\phi), \sin(n\theta)\sin(n\phi), cos(n\theta))$$

donde

$$r = \|p\| \qquad \theta = atan2(\|(x,y)\|, z) = \arccos(\frac{z}{r}) \qquad \phi = atan2(y, x)$$

John C. Hart en su artículo 'Ray tracing Deterministic 3D fractals' asegura que para la familia cuadrática, tenemos la siguiente aproximación:

$$d(z) = \frac{|f^n(z)|}{2|f'^n(z)|}\log|f^n(z)|$$

Mandelbulb

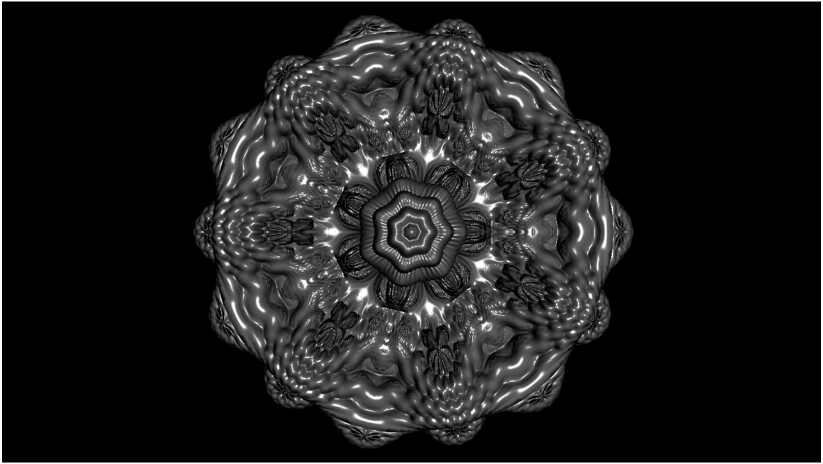


Figura: [Python] Mandelbulb

Mandelbulb

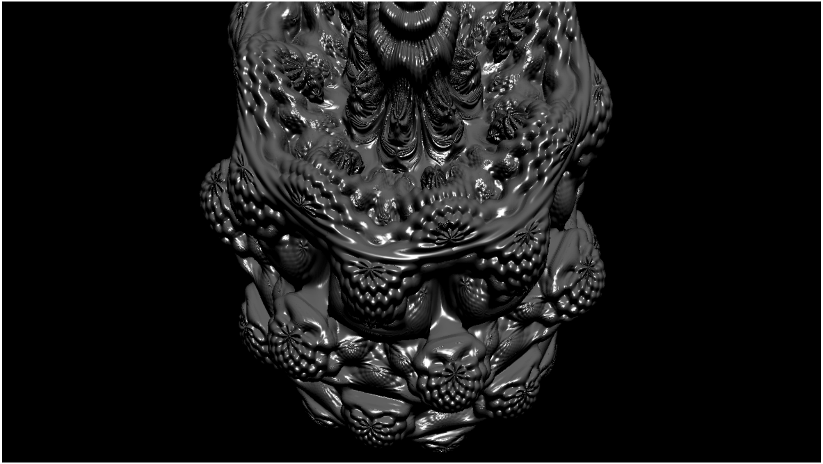


Figura: [Python] Mandelbulb

Ray marching

Juliabulb

Los cuaterniones de Julia se construyen según el mismo principio que el conjunto de Julia tradicional. Utiliza números complejos de cuatro dimensiones en lugar de complejos de dos Un cuaternión tiene dos componentes complejas más y se escribe como $q = r + ai + bj + ck$ donde $r, a, b, c \in \mathbb{R}$. Existen relaciones un poco más complicadas entre i, j y k.

$$i^2 = j^2 = k^2 = -1$$

$$ij = k \quad jk = i \quad ki = j$$

$$ji = -k \quad kj = -i \quad ik = -j$$

UNIVERSIDAD DE GRANADA

Cuaternión de Julia

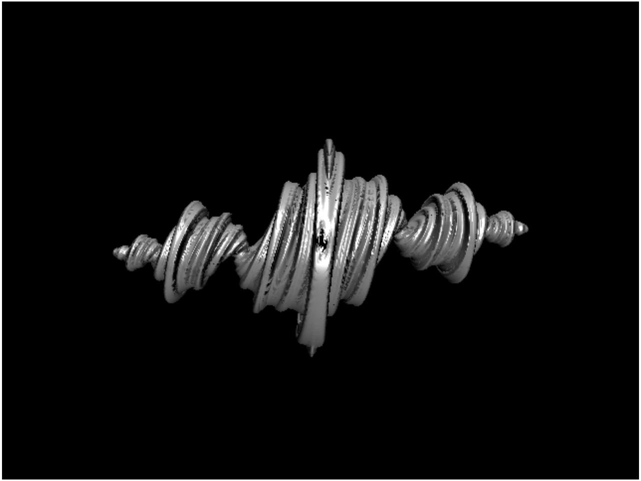


Figura: [Python] Cuaternión de Julia $c = (-1, 0{,}2, 0, 0)$

Cuaternión de Julia

Figura: [Python] Cuaternión de Julia $c = (-0{,}125, -0{,}256, 0{,}847, 0{,}0895)$

UNIVERSIDAD DE GRANADA

Mandelbox

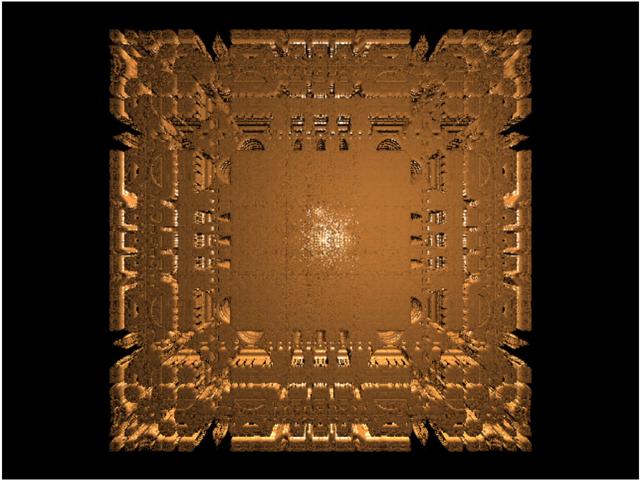


Figura: [Python] Mandelbox

Optimización de renderizado

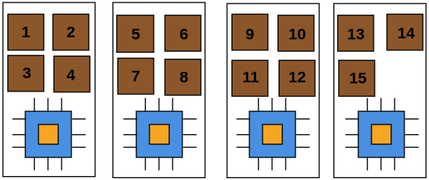


Figura: [Mathcha] División de 15 filas de píxeles entre 4 núcleos

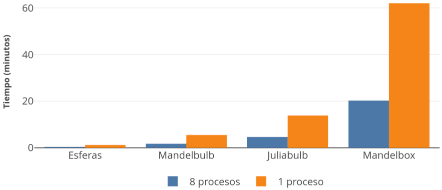


Figura: [Mathcha] Comparativa de tiempos de renderizado

FIN