Mandelbrot Rendering

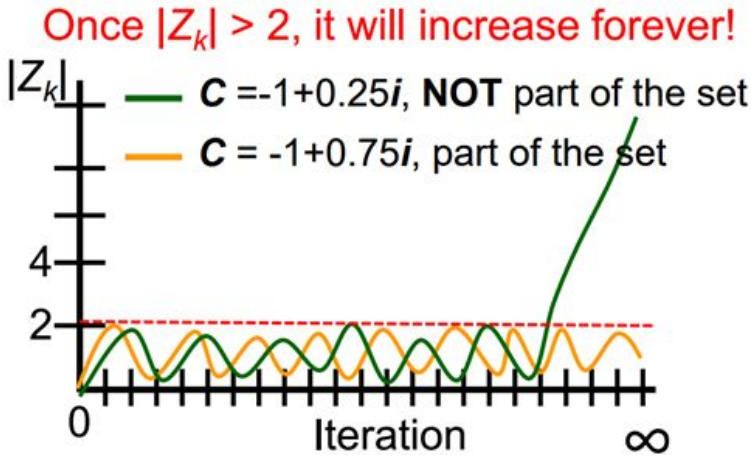
Introduction to Parallel Computing - 2020 Spring

Index

1. Mandelbrot Set
2. Mandelbrot Set Visualization
3. Mandelbulb
4. Mandelbulb Visualization
5. Goal
6. Requirements

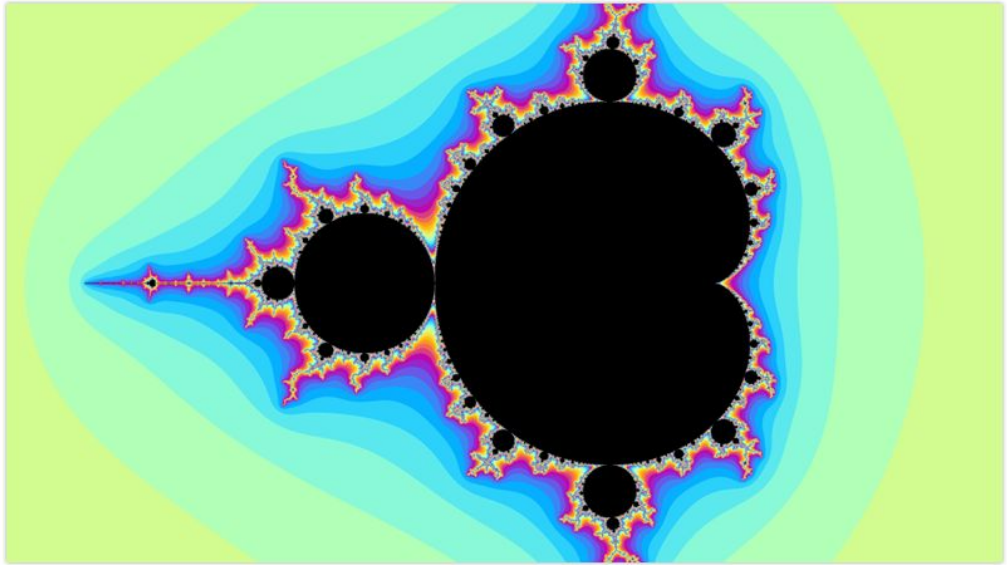
Mandelbrot Set

- A set of complex numbers $\mathbb{C}$
  - for every complex number $c \in \mathbb{C}$, under iterations of quadratic map $Z_{k+1} = Z_k + c$ remain bounded
  - i.e, it satisfies for every Z_k, where k is zero or a positive integer
    - $Z_{k+1} = Z_k + c$
    - $Z_0 = 0$
    - $|Z_k| \leq 2$



Mandelbrot Set Visualization

- Convert each pixel to the corresponding coordinates on the complex plane
- Plug into the equation repeatedly until |Zk| > 2
- Color the pixel according to the iteration count



Mandelbulb

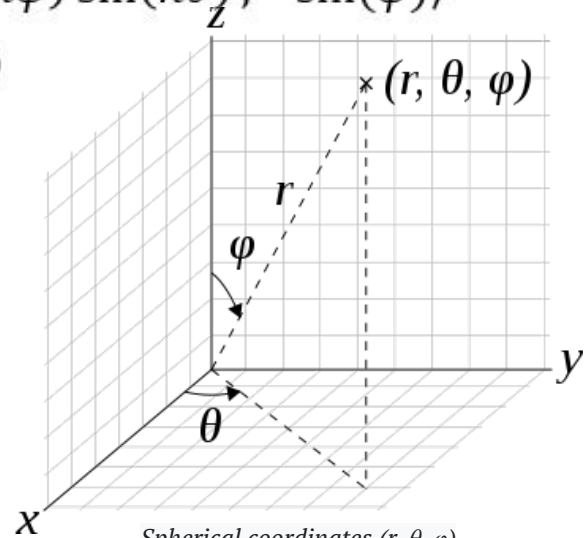
- A 3D fractal use [spherical coordinates] to represent its 3D space
- In this assignment, we refer to power 8 mandelbulb

$$v_{k+1} = v_k^8 + C$$

$$v = \langle x, y, z \rangle \text{ in } \mathbb{R}^3,\ v^n := r^n \langle \cos(n\theta)\cos(n\phi), \cos(n\phi)\sin(n\theta), -\sin(\phi) \rangle$$

- $r = \sqrt{x^2 + y^2 + z^2},\ \theta = \arctan\left(\frac{y}{x}\right),\ \phi = \arctan\left(\frac{z}{r}\right)$

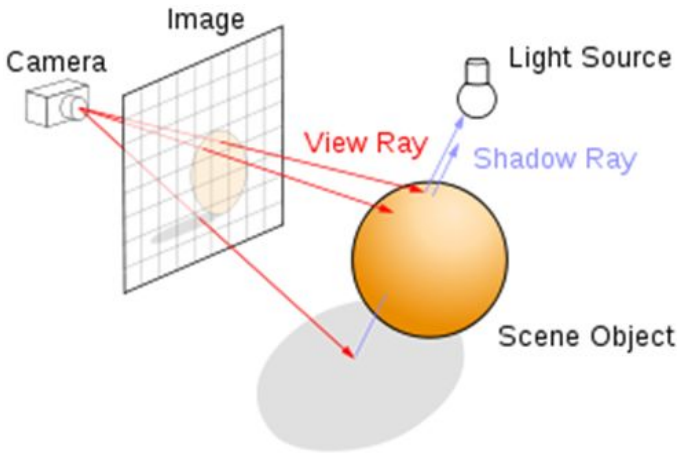
- e.g. C = <1, 2, 3>, v_0 = <0, 0, 0>
  - r = √14

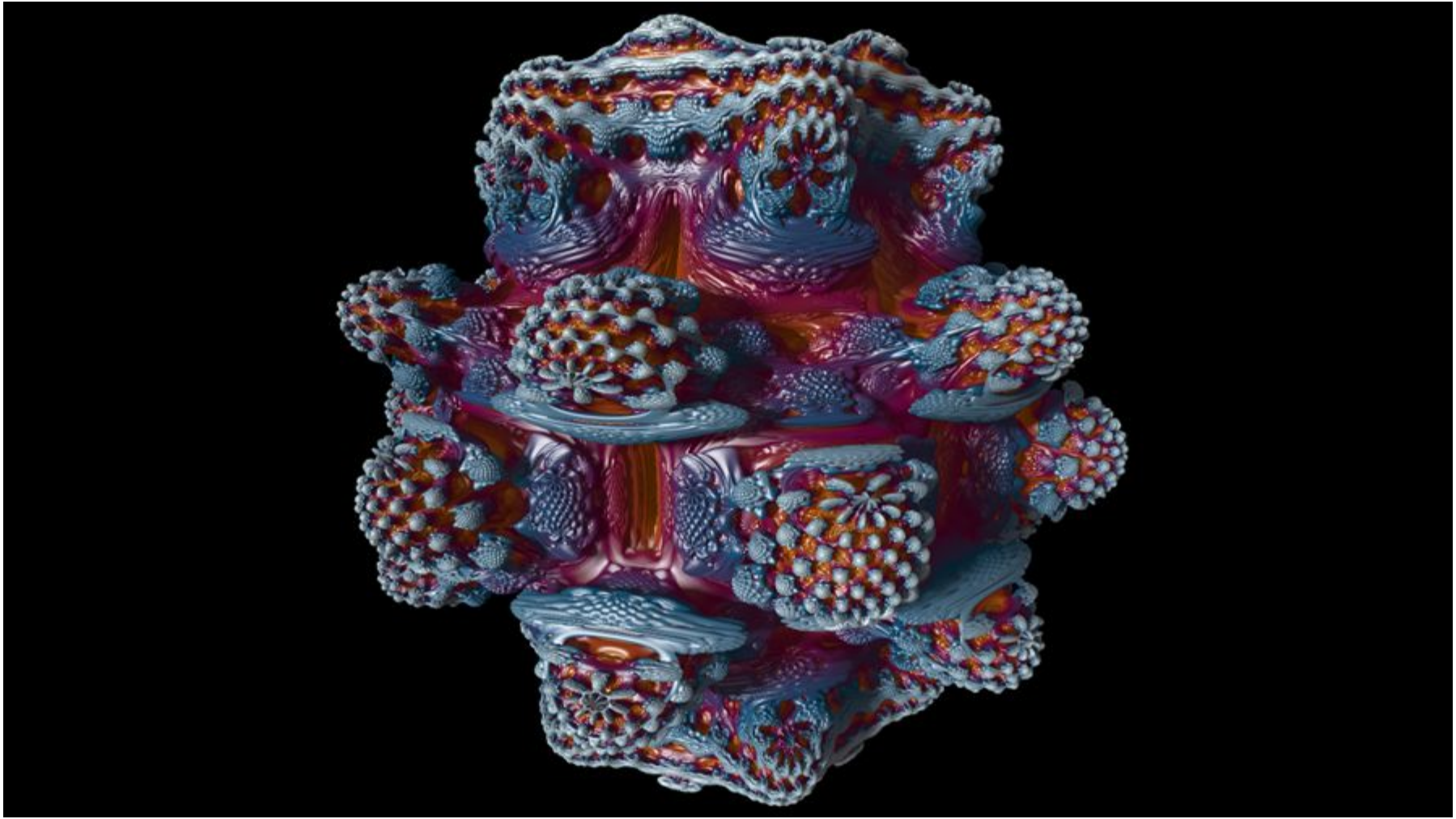


Spherical coordinates (r, θ, φ)

Mandelbulb Visualization - Ray Marching

- A kind of ray tracing algorithm.
- Casting a 3D ray for each screen pixel, uses a mathematical function called Distance Function (or Distance Estimator) to verify if the ray intersects with any objects.





Goal

- TA gives the sequential code for visualizing mandelbulb.
- You are asked to parallelize with MPI & OpenMP

Requirements

- Submit below files to ilms directly:
  - hw2.cc - the source code of your implementation
  - build.ninja - optional. Submit this file if you want to change the build command
  - report.pdf - your report
- Please refer to IPC20_HW2.spec.pdf for details
- Deadline
  - 2020/04/13 (Mon.)