Exploring Cellular Automata: From Simple Rules to Complex Patterns

Introduction

Cellular automata are discrete computational models composed of a grid of cells, each capable of transitioning between states based on simple rules defined by neighboring cells. Originally introduced in the 1940s by Stanislaw Ulam and John von Neumann, these models offer a fascinating glimpse into how complexity can emerge from simplicity. This project explores both elementary and complex cellular automata, focusing on visual behavior patterns and their underlying rules.

Motivation

The key motivation for this exploration lies in understanding how simple, deterministic rules can produce a wide variety of behaviors — from stability and repetition to randomness and complexity. Applications of such models span cryptography, artificial life, complex systems, and biological modeling. We aimed to:

- Examine how rule-based systems evolve over time.
- Discover emergent complexity in deterministic setups.
- Compare simple elementary automata with a more dynamic model (Brian's Brain).

Problem Statement

We investigate:

- Ten elementary cellular automaton rules, especially Rule 110 and Rule 4.
- Which rules lead to complex vs. simple patterns?
- How behavior types vary across systems.
- How Brian's Brain, a more biologically inspired automaton, behaves differently.

Related Work

Elementary Cellular Automata:

- Defined by Stephen Wolfram's 256 rules.
- Rules fall into four behavior classes:
    1. Uniform
    2. Stable/Repetitive
    3. Random (chaotic)
    4. Complex (localized interactions)

Brian's Brain:

- A 2D automaton using Moore neighborhood (8 neighbors).
- Each cell transitions through 3 states:
    - OFF → ON (if exactly two ON neighbors)
    - ON → DYING
    - DYING → OFF
- No static patterns; constant wave-like motion.

Approach

We implemented two systems in Python using Pygame and NumPy:

Elementary Cellular Automaton

- Converts a rule number (0–255) to an 8-bit binary string.
- Simulates 1D evolution over time with a single active cell in the center.
- Outputs patterns across 600 rows.
- Users can press ↑/↓ to cycle through rules and "S" to save the output image.

Brian's Brain Cellular Automaton

- Initializes a 2D grid randomly (80% OFF, 20% ON).
- Each frame updates all cells based on neighbors.
- Used Pygame for visual animation and imageio to export GIFs.
- Included ability to restart or save simulation via keyboard input.

Results

Elementary CA Observations:

- Most rules lead to:
    - Uniform patterns (Class 1)
    - Repetitive/stable behavior (Class 2)
- ~14% showed complexity or nesting.
- ~10% displayed randomness (Class 3, e.g., Rule 30).
- <5% showed Class 4 behavior (localized structures, e.g., Rule 110).

Brian's Brain Results:

- Patterns never stabilize; remain in perpetual motion.
- Resembles chaotic, self-sustaining neural or wave activity.
- Visualized with distinct colors: cyan (ON), orange (DYING), black (OFF).
- Demonstrated effective propagation and feedback cycles.

Conclusion

This project demonstrates how simple computational rules can lead to a surprising variety of emergent behaviors. Elementary Cellular Automata revealed how different rules drive pattern formation — from fractals to randomness. Brian's Brain illustrates dynamic, biologically inspired complexity that never converges to stasis.Both simulations were implemented in Python with real-time graphical output and exportable visualizations. Through this project, we gained a deeper appreciation of complexity theory, deterministic chaos, and the intersection of art and computation.