Illuminated Drone Show Simulation

Presented by Ramaz Botchorishvili

Kutaisi International University

December 12, 2024

Project Overview: Orchestrating a Digital Sky

This project simulates the intricate choreography of 1200 drones forming complex illuminated shapes in a virtual environment. We leveraged numerical methods to tackle a fascinating real-world engineering challenge, focusing on precision, efficiency, and visual appeal.

1200 Drones

Massive swarm simulation

Shape Formation

Complex light patterns

Numerical Methods

Applied real-world physics

Core Subtasks:

01

Static Shape Transition

02

Sequential Static Transitions

03

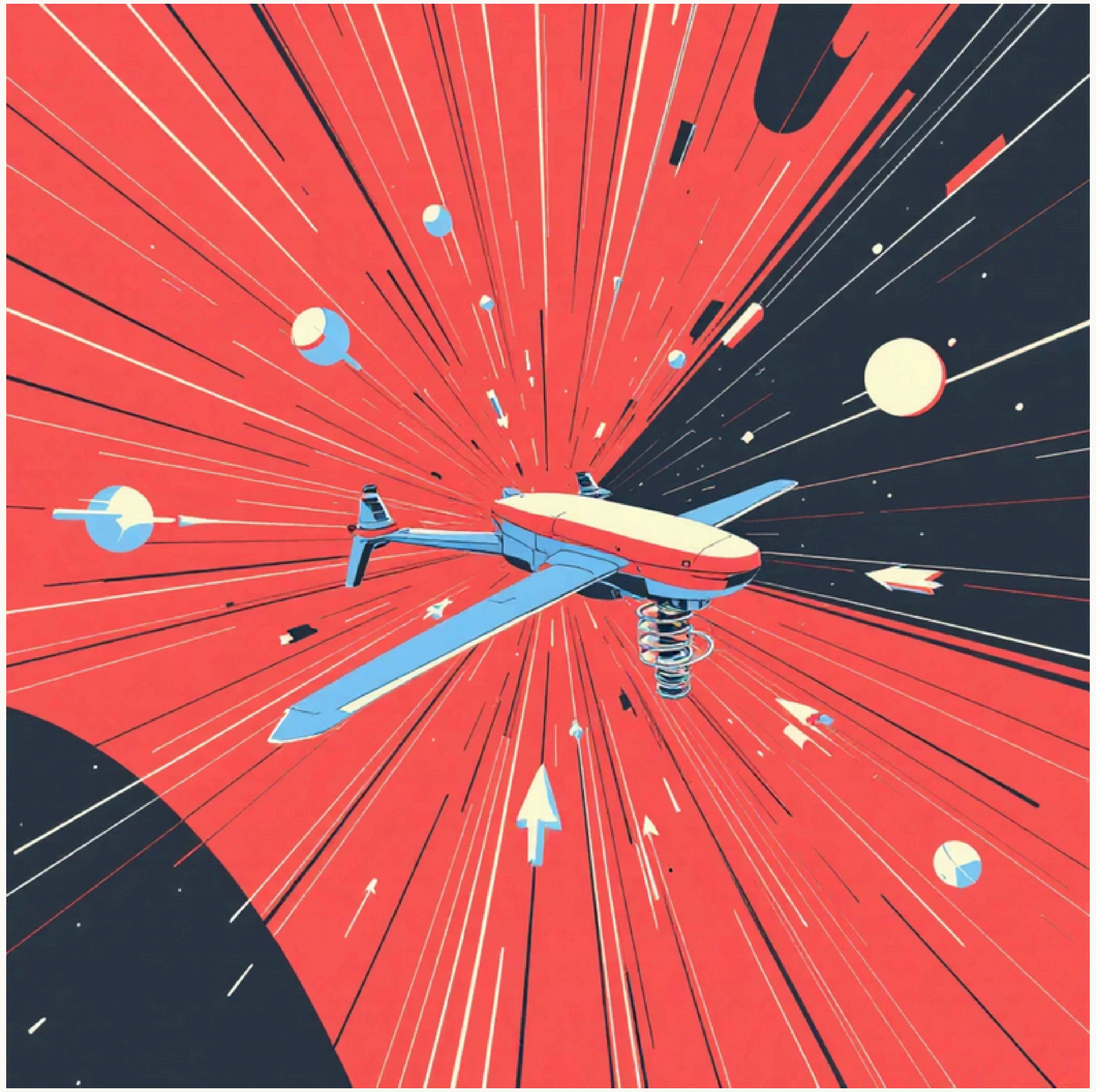
Dynamic GIF Tracking

Mathematical Model: The Drone's Dynamics

Each drone's movement is governed by a robust Initial Value Problem (IVP), integrating forces that dictate its velocity and position.

$$\frac{dx}{dt} = v \quad \min(1, \frac{v_{max}}{|v|})$$

$$\frac{dv}{dt} = \frac{1}{m}[k_p(T - x) + f_{rep} - k_d \quad v]$$



Key Force Components:

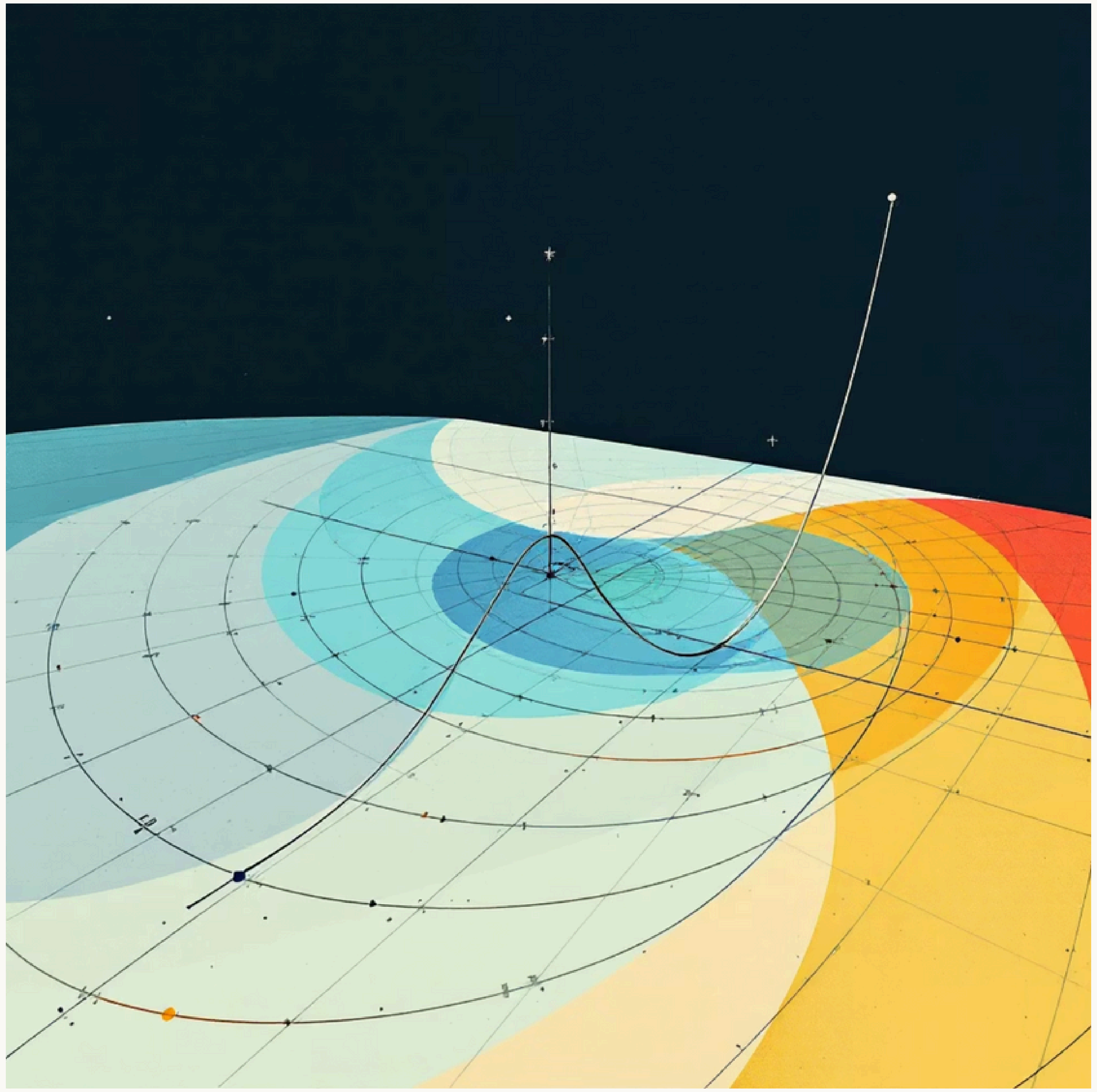
- **Spring Force:** Draws drones towards target positions (T-x).
- **Damping:** Opposes velocity to stabilize movement (k_d · v).
- **Repulsion:** Prevents collisions with other drones (f_rep).

RK4 Integration: Precision in Motion

To accurately solve the IVP, we employ the 4th-order Runge-Kutta method, known for its stability and precision in numerical integration.

$$y_{next} = y + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

- **k1:** Slope at the beginning of the interval.
- **k2:** Slope at the midpoint, based on k1.
- **k3:** Slope at the midpoint, based on k2.
- **k4:** Slope at the end of the interval, based on k3.



Error: The RK4 method provides an error of O(h^4), ensuring high accuracy over time steps (h).

Collision Avoidance: Maintaining Formation Integrity

A critical aspect of a drone swarm is collision avoidance. We implemented a repulsive force model, inspired by electrostatics, to ensure drones maintain safe distances.

$$f_{rep} = k_{rep} \frac{(x_i - x_j)}{|x_i - x_j|^3}$$

- **Inverse-Square Law:** The repulsive force is inversely proportional to the cube of the distance between drones, preventing close proximity.
- **Safety Radius:** A defined safe zone around each drone (Rsafe = 4 units) triggers the repulsive force, ensuring clear space.

Spatial Hashing: Efficient Neighbor Discovery

To handle 1200 drones efficiently, a brute-force $O(n^2)$ collision check is infeasible. Spatial hashing dramatically optimizes neighbor queries to $O(n)$, enhancing performance.

Hash Function:

$$h(x, y) = \quad (\text{floor}(x/c), \text{floor}(y/c))$$

- **Grid Partitioning:** The simulation space is divided into a grid, with each cell identified by its hashed coordinates.
- **Neighborhood Check:** Each drone only checks for collisions within its own cell and the 8 surrounding cells (a 3x3 neighborhood).

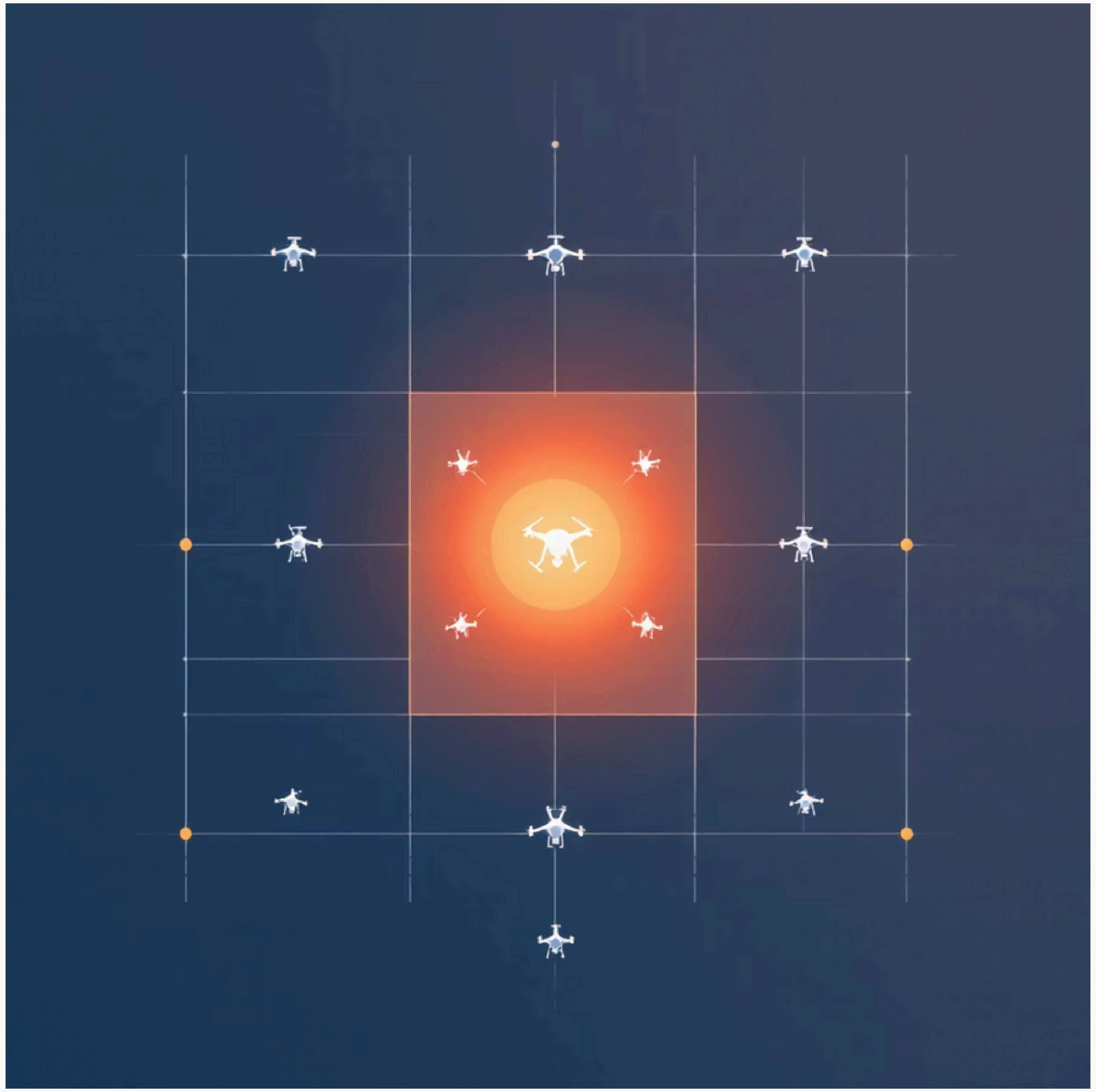


Image Processing Pipeline: Extracting Target Shapes

Before drones can form shapes, target images must be processed to extract precise point data. This pipeline transforms raw images into actionable coordinates.

1

Gaussian Blur

Noise reduction

2

Sobel Operators

Gradient detection

3

Canny Edge

Precise edge detection

4

Point Extraction

Convert edges to drone targets

This multi-step process ensures high-quality edge detection, providing accurate target points for the drones to converge upon.

The 3 Subtasks: From Grid to Dynamic Art

The project culminates in three distinct phases, each demonstrating different aspects of drone control and animation.

1

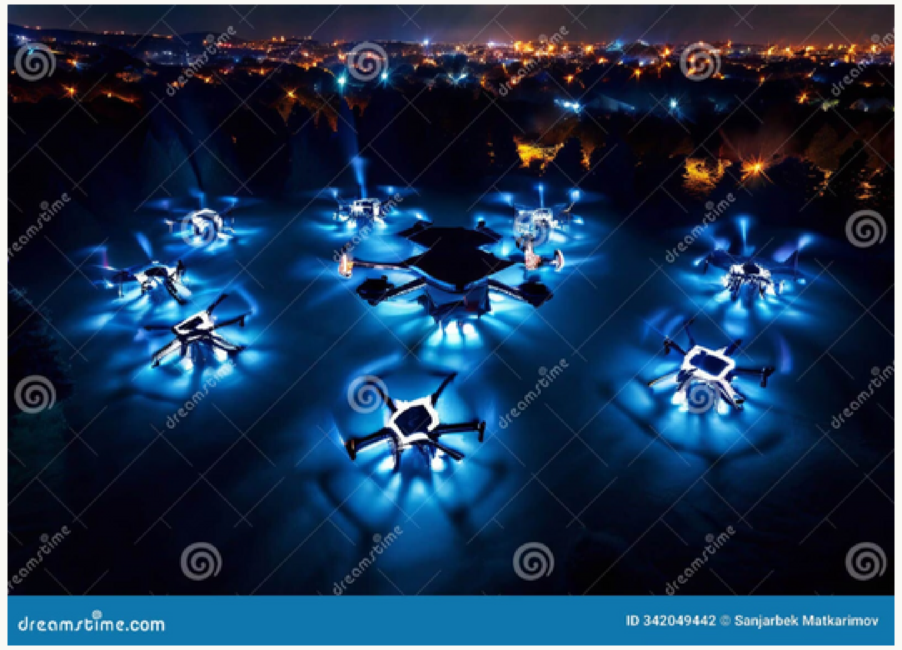
Phase 1: Grid → Handwritten Name

110 frames: Drones transition from a uniform grid to form a predefined name.

2

Phase 2: Name → Greeting Image

110 frames: The name shape smoothly morphs into a greeting image.



3

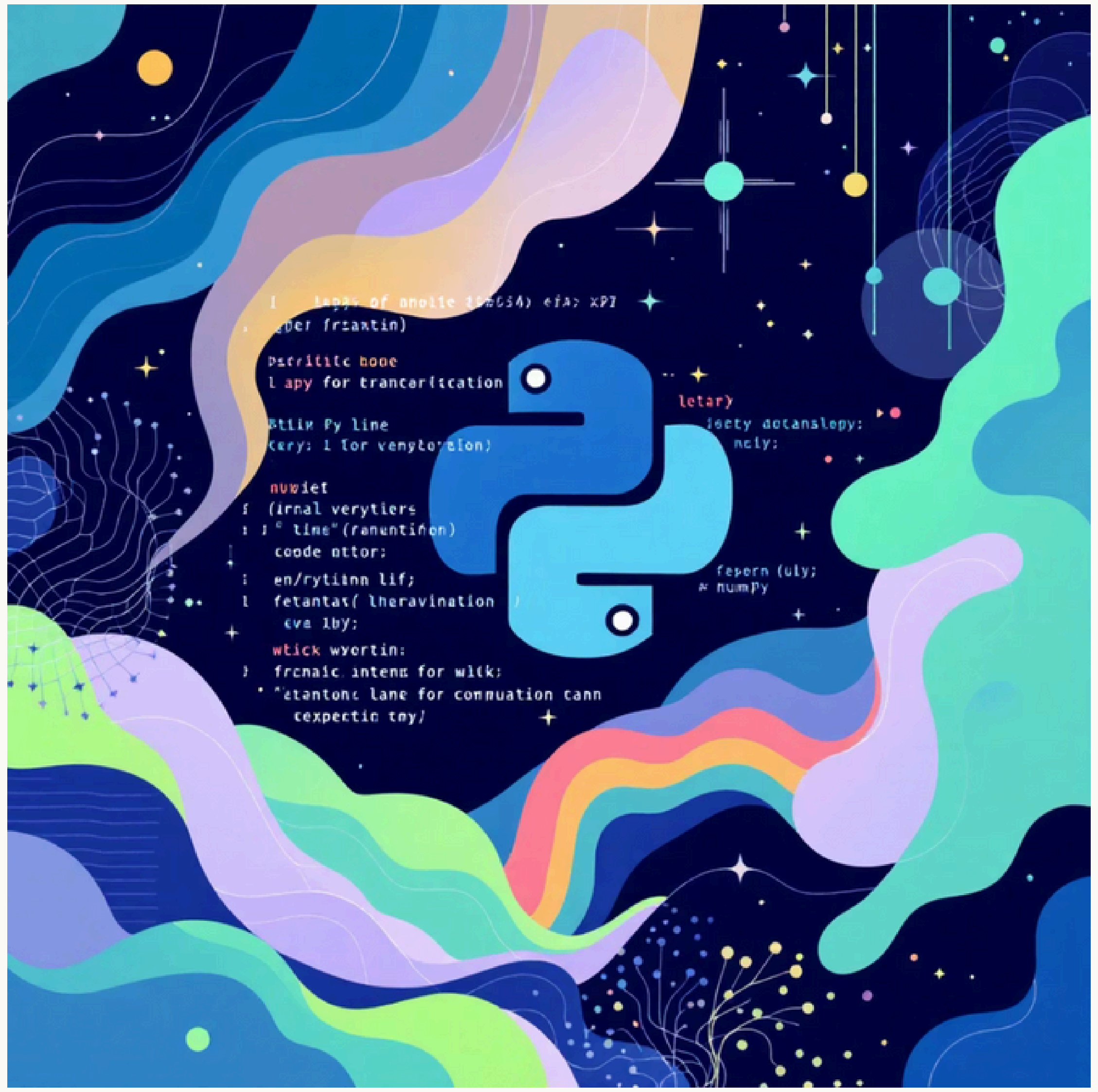
Phase 3: Dynamic GIF Tracking

246 frames: Drones follow the dynamic animation of a GIF, demonstrating real-time tracking capabilities.

Implementation Details: Code & Configuration

The entire simulation is built from the ground up using fundamental numerical programming principles.

- **Language:** Python
- **Libraries:** NumPy (pure mathematical operations)
- **Algorithm Philosophy:** Custom implementations for all algorithms, avoiding high-level SciPy functions.
- **Scale:** 1200 drones across 466 total frames.
- **Output:** ~15-second video at 30 FPS.



Key Parameters:

M	0.5	Drone mass
kp	1.0	Spring constant
kd	0.2	Damping constant
krep	1000	Repulsion constant
vmax	5.0	Max velocity
dt	0.01	Time step

Results & Conclusion: A Symphony of Light

The simulation successfully orchestrates 1200 drones, demonstrating smooth transitions, effective collision avoidance, and precise shape formation.

- Integrated IVP model with RK4 for accurate physics.
- Successfully implemented custom numerical algorithms without external shortcuts.
- Real-time collision avoidance mechanisms ensure a flawless display.

> This project highlights the power of numerical programming in bringing complex visual concepts to life.

Questions?

GitHub Repository: github.com/devichikovani/np-final-project