Cat Swarm Optimization (CSO)

Project Report

By

Sharaiz Ahmed [57288]

Supervised

by

Muhammad Usman Shariff

Analysis of Algorithm

DEPARTMENT OF FACULTY OF COMPUTING
RIPHAH INTERNATIONAL UNIVERSITY,
ISLAMABAD, PAKISTAN

TABLE OF CONTENTS

1. INTRODUCTION

Cat Swarm Optimization (CSO) is a swarm intelligence-based metaheuristic algorithm introduced by Shu-Chuan Chu in 2006. Inspired by the natural behavior of cats, CSO divides the actions of cats into two modes: seeking (resting and observing) and tracing (pursuing a target). This dual-mode strategy mimics the natural instincts of cats and allows the algorithm to explore and exploit the search space efficiently. CSO has proven to be an effective algorithm for solving continuous and discrete optimization problems due to its simplicity and low computational cost.

2. PROBLEM STATEMENT

The growing use of drones in logistics demands efficient algorithms to plan optimal delivery paths. The aim of this project is to optimize the travel path of a drone to reach a fixed delivery location efficiently. The challenge is to compute the most efficient trajectory from the drone's starting position to a fixed delivery location while minimizing the distance and computation time. This project uses Cat Swarm Optimization to simulate this process, treating each drone as a "cat" in the swarm that adapts its movement based on the optimization strategy.

3. ALGORITHM OVERVIEW

In CSO, each cat is a potential solution with a position and a velocity. The algorithm operates by dividing the population into two groups: cats in seeking mode and cats in tracing mode.

- **Seeking Mode** is the exploration phase where cats look for better positions by evaluating several candidate positions nearby.

- **Tracing Mode** is the exploitation phase where cats move toward the best-known solution.

Key parameters include:

- Number of cats (population size)
- Seeking memory pool (number of candidate positions)
- Mixing ratio (proportion of cats in seeking vs. tracing mode)
- Seeking range of the selected dimension

Over multiple iterations, the cats adapt their positions based on these behaviors, gradually converging to the optimal or near-optimal solution.

4. METHODOLOGY

The methodology adopted in this project follows a structured approach based on the principles of algorithm design and empirical analysis:

4.1. PROBLEM FORMULATION

We define the objective function as minimizing the Euclidean distance between the drone's current position and the fixed delivery point (7, 5).

4.2. PARAMETER INITIALIZATION

- Total cats (drones): 20
- Iterations: 100
- Seeking mode: 100% (simplified implementation)
- Initial positions: Randomized within a 10x10 grid
- Velocity: Not applied in seeking-only version

4.3. FITNESS FUNCTION

Each cat evaluates its fitness as:
Fitness = sqrt((x - 7)² + (y - 5)²)
where `(x, y)` is the current position of the drone.

4.4. MOVEMENT STRATEGY

In seeking mode:

- Multiple nearby positions are generated.
- The best one (shortest distance) is selected as the new position.
- The position is updated only if the new candidate is better.

4.5. TERMINATION CRITERIA

The algorithm runs for a fixed number of iterations (100) or stops early if the best fitness remains unchanged over successive iterations.

4.6. PERFORMANCE EVALUATION

The results are evaluated by comparing initial average distances with final optimized distances, and visual tracking of path convergence is included.

5. CODE IMPLEMENTATION

The implementation uses C++ for a simple drone delivery optimization task. Each drone tries to move toward a fixed delivery point at (7, 5) and includes the following components:

- A Drone class representing each cat with x, y coordinates and fitness
- A function to initialize random positions
- A seeking function to generate candidate moves
- A main loop iterating over updates and recalculating fitness

6. COMPLEXITY ANALYSIS

TIME COMPLEXITY:

O(N × I × S)

- N = number of cats
- I = number of iterations
- S = number of candidate solutions per cat in seeking mode

SPACE COMPLEXITY:

O(N)

Only the position and fitness values are stored for each cat.

This ensures scalability in small to medium-size search spaces and allows for fast convergence in real-time applications.

7. REAL-WORLD APPLICATION: DRONE DELIVERY OPTIMIZATION

Drones are becoming increasingly integral to logistics for delivering small packages, especially in remote or urban congested areas. In such scenarios, real-time optimization algorithms are needed to adapt flight paths and reduce energy consumption. Delivery drones must quickly adapt their path due to wind, air traffic, or dynamic constraints. CSO enables:

- Fast recalculations of optimal paths
- Adaptability in constrained environments
- Lightweight computation for embedded systems

CSO continuously adjust the path, helping drones dynamically navigate to their destinations. Though simple, this implementation could be extended to include 3D movement, energy constraints, and obstacle avoidance for more realistic use.

8. LIMITATIONS

While CSO shows promise, the current version has the following limitations:

- Only seeking mode is implemented; no tracing mode for global convergence
- Assumes a static environment with no dynamic obstacles like obstacles or wind
- Only a single delivery point is used
- Scalability is limited to small populations due to simplistic design
- Not inherently suitable for large-scale 3D navigation without modification

Future work may include hybridization with other algorithms, multi-point optimization, and environmental modeling.

9. CONCLUSION

Cat Swarm Optimization (CSO) is a smart and simple algorithm that can help solve problems where the best answer isn’t easy to find through normal methods. In this project, we used CSO to help a drone find the shortest path to a delivery point. By simulating how cats behave, either sitting still and watching or chasing something, we were able to show how drones (acting like cats) can move closer to their target over time.

Even with only the basic part of the algorithm (the seeking behavior), the drones were able to find a better path and reduce the distance to the delivery point. This

shows that CSO can work well for small and simple problems, especially when quick decisions are needed and there isn't much computing power available.

However, the current version is limited. It doesn't include all the features of the full CSO algorithm, like the tracing mode that helps find even better paths. It also doesn't deal with real-world issues like obstacles or multiple delivery points. These improvements would be important if we wanted to use this in more complex or real situations.

In short, this project showed that CSO works well for basic drone delivery path planning, and with some upgrades, it could become a useful tool in real-life systems like smart delivery drones or robots.

10. REFERENCES

 Chu, S.-C., Tsai, P.-W., & Pan, J.-S. (2006). *Cat Swarm Optimization*. In Proceedings of the 9th Pacific Rim International Conference on Artificial Intelligence (pp. 854–858). Springer.

 Kennedy, J., & Eberhart, R. (1995). *Particle Swarm Optimization*. Proceedings of ICNN'95 - International Conference on Neural Networks, 4, 1942–1948. IEEE. *(Referenced for algorithm comparison)*

 Yang, X. S. (2014). *Nature-Inspired Optimization Algorithms*. Elsevier. *(Used for methodology and comparison context)*

11.GITHUB LINK

https://github.com/Sharaiz333/Analysis-of-Algorithm/tree/main/Semester%20Project