International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Efficient Shortest Path Finding Using Chaotic Fly Consonance Optimization Algorithm

[1]Mrs. M. Kavitha, [2]Dr.G. T. Prabavathi

Submitted: 22/05/2024 **Revised**: 12/07/2024 **Accepted**: 22/07/2024

Abstract: This study presents an effective method for determining the shortest route for vehicles in urban traffic situations by combining the Chaotic Fly Consonance Optimization (CFCO) algorithm with the Deep Deterministic Policy Gradient (DDPG) algorithm for traffic flow estimation. By taking into account real-time traffic conditions and optimizing routes appropriately, the technique attempts to improve the overall performance of vehicle navigation systems. In the first stage, the system receives input in the form of traffic signal pictures ranging from 0 to 10, as well as the planned destination route. Following that, the deep deterministic policy gradient (DDPG) method is used to evaluate traffic flow, taking use of its model-free off-policy nature for continuous action reinforcement learning. This stage entails fine-tuning hyperparameter to produce the best possible outcomes in anticipating traffic conditions. Following the assessment of traffic flow, the suggested chaotic fly consonance optimization (CFCO) method is used in the third stage to discover the vehicle's shortest route. The CFCO method is used for global optimization and was inspired by the chaotic behavior of flies in nature. This stage entails using the algorithm's chaotic nature to effectively search the solution space and locate optimum pathways. The last phase displays the shortest route found by combining DDPG for traffic flow assessment and CFCO for path optimization. The approach is assessed based on its capacity to adapt to changing traffic circumstances and propose effective routes, eventually leading to better vehicle navigation in metropolitan areas.

Keywords: *Chaotic Fly Consonance Optimization, Deep Deterministic Policy Gradient, Efficient Path Finding, Traffic Flow.*

1. Introduction

When talking about ways to reduce traffic congestion on roads, "smart" means using data analysis and modern technologies. Heavy traffic makes it impossible for the passenger to reach their destination on time in a smart city [1]. In order to alleviate traffic, graph theory algorithms are used to determine the road's structure; in the event that there is damage or construction, the algorithms will keep an eye out and solicit feedback from drivers [2]. They claim that an upcoming traveler will see the outcome. For example, if there is an obstruction on the road, the next vehicle will be rerouted to take the shortest alternative route [3]. Even though it would be the easiest option for the passenger, the shortest path here wouldn't get them to the important shortest route, which would provide them with a seamless flow [4]. So, easing traffic benefits other drivers who don't have to deal with heavy traffic. The passenger may avoid traffic jams based on prior findings [5].

Efficient vehicle navigation is crucial in modern urban settings for reducing trip times, improving transportation systems generally, and alleviating traffic congestion [6]. Optimal route planning in traffic situations that are constantly changing has prompted the use of AI methodology and sophisticated optimization techniques [7]. To tackle the problems with vehicle pathfinding, this study suggests a new method that merges two algorithms: one for assessing traffic flow, the DDPG algorithm, and another, the Chaotic Fly Consonance Optimization (CFCO) algorithm, to find the shortest and most efficient routes [8]. The intricacy of urban road networks necessitates smart and adaptable systems to guide vehicles through traffic patterns and congestion, which are becoming more complicated as cities expand [9]. Insufficient real-time traffic situation adaptation is a common problem with traditional route planning approaches, resulting in less-than-ideal pathways and longer trip times [10]. Our study seeks to address these difficulties by using cutting-edge AI and optimization techniques. The goal is to improve vehicle routes dynamically according to current traffic circumstances [11].

To evaluate and forecast traffic flow, the suggested technique incorporates the DDPG algorithm, a model-free off-policy reinforcement learning algorithm [12]. Because vehicle traffic in metropolitan areas is inherently continuous, this integration enables continuous action decision-making [13]. Furthermore, a

[1]*Research Scholar, Department of Computer Science, Gobi Arts and Science College, and Assistant Professor, Department of Computer Science, KG College of Arts and Science, Coimbatore. E-Mail: kavithalakshmi.m@gmail.com*
[2]*Corresponding Author: Associate Professor, Department of Computer Science, Gobi Arts and Science College, Gobi. E-Mail: gtpraba@gmail.com*

global optimization method that draws inspiration from the naturally chaotic behavior of flies is presented as the Chaotic Fly Consonance Optimization (CFCO) algorithm [14]. The CFCO method identifies the most efficient paths by successfully exploring the solution space using chaotic search techniques [15]. A new framework that combines the capabilities of AI-based traffic flow assessment and chaotic optimization for pathfinding [16-17] is presented in this study as a means to close the gap between existing navigation systems and the growing dynamics of urban traffic. Following this, we will go into the specifics of how to put the suggested method into action, starting with importing pictures of traffic signals and route data, then running the DDPG algorithm, and last, using the CFCO algorithm to find the best possible route [18]. We expect the findings to show that this integrated technique is effective in making urban vehicular navigation efficient and flexible [19–20].

The main contribution of the paper is:

- Input Traffic Signal Images 0-10, path
- Checking Traffic Flow using DDPG (Deep Deterministic Policy Gradient (DDPG) with Hyper Parameters
- Apply CFCO algorithm to finding the Shortest Path
- Shortest Path Result

The remainder of the paper is structured as follows. Section II provides an evaluation of the current status of research on efficient shortest route finding algorithms. In Section III, the reasoning behind the proposed method is explored in additional detail. The experimental data are presented and discussed in Section IV. In Section V, we wrap up the study by talking about possible future research directions.

1.1 Motivation of the paper

We need more sophisticated solutions to the problems caused by urban traffic congestion since conventional route design approaches aren't cutting it anymore. Incorporating reinforcement learning and chaotic optimization into intelligent transportation systems is the driving force behind this study. The goal is to create a method that can change and adapt to the environment, taking into account traffic circumstances in real-time, optimize routes effectively, and make automobile navigation systems far more effective in complicated urban settings. Congestion relief, faster travel times, and an improved experience for city commuters are the end goals.

2. Background Study

Alves, D. et al. [1] three new label-setting methods for the Shortest Path issue were introduced and tested in this research: The single-threaded methods SP1 and SP2 use the graph's known properties to label nodes as fixed, preventing heap inserts and removals; the multi-threaded technique ParSP2 takes use of the chances for parallelism provided by SP1 and SP2.

Candra, A. et al. [3] the results of the studies prove that both Dijkstra's and A* were capable of finding the shortest route. But sometimes the two algorithms come up with distinct paths, which mean the overall distance was varied too. In addition, A* has a shorter running time than Dijkstra's; although Dijkstra's averages 7.719 ms, A* clocks in at 4.406 ms. This follows the A* principle, which chooses the site using the optimal heuristic value. Lastly, it was believed that Dijkstra's and A* algorithms have the same complexity.

Gbadamosi, O. A., & Aremu, D. R. [7] these authors research propose an alternative to traditional Dijkstra's method for determining shortest paths when the costs of doing so were so high that the benefits of using it were outweighed. Using a 40-node graph, the new method was put into action. This study was a continuation of an ongoing research project; the next phase will focus on fully implementing the algorithm and calculating its complexity from analytical and numerical perspectives.

Kučera et al. [9] By using graph theory to pre-process historical and real-time traffic data with different input types, the shortest route*search efficiently finds the shortest path in regions with high levels of traffic congestion. Prior to processing each step, the beginning node focuses on locating the closest desired objective. The shortest route * search algorithm's primary benefit was that it avoids squandering time on unnecessary nodes.

Liu, H. et al. [11] the top-k shortest pathways with diversity (KSPD) were the focus of this paper's research. Once the issue has been clearly defined, the author demonstrates that the KSPD problem was NP-hard. Then, in cases when diversity was not necessary, the author provide a generic greedy framework for the KSPD issue that can handle other similarity functions as well as the KSP problem.

S. Santos et al. [13] For any given quantum network source node, this article presents an efficient method that determines all the nodes with whom it may form a quantum connection with a certain end-to-end fidelity requirement, as well as the shortest way to reach each of those nodes.

Shen, L. et al. [15] An new approach was presented and used in this study to tackle the reliable shortest path-finding problem in road networks with related link travel time, delays at signalized junctions, and traffic signals, as well as their correlations. Whether a road network was

crowded or not, these authors model can capture both conditions. Two separate stochastic processes were used to simulate these two separate cases, drawing on previous research. In other words, the author assumes that saturation flows were lognormally distributed, and journey times on crowded highways follow normal distributions

Toan, T. et al. [17] these authors research introduces a method for using visibility-graph for robots to determine the shortest route between two points in a two-dimensional environment while avoiding fixed barriers.

Lei et al. [23] introduce an enhancement to this algorithm by integrating chaotic systems, resulting in the Chaotic Fruit Fly Optimization Algorithm (CFOA). The FOA models this behavior through iterative steps of searching and updating positions based on a combination of olfactory and visual cues. It has been applied to various optimization problems, demonstrating simplicity and effectiveness.

2.1 Problem definition

There is a major issue with contemporary transportation in the form of urban traffic congestion and the inefficiency of conventional route design techniques. In order to optimize automobile navigation in urban environments, this study presents a novel approach to the problem. Finding the most direct and effective routes for vehicles while taking current traffic circumstances into account is the main challenge. Reducing travel times and improving navigation in urban contexts are two of the main goals of the challenge, which also requires finding solutions to the constraints of current systems and being able to react to changing traffic conditions.

3. Materials and Methods

Using the Chaotic Fly Consonance Optimization (CFCO) algorithm, we developed an effective shortest pathfinding system for automobiles. The materials and methodology applied in this study are outlined in this section. Utilizing photos of traffic signals, the DDPG algorithm for evaluating traffic flow, and the CFCO algorithm for determining the best route are all part of the process. The efficient shortest path finding using chaotic fly consonance optimization algorithm model has represented at figure 1.

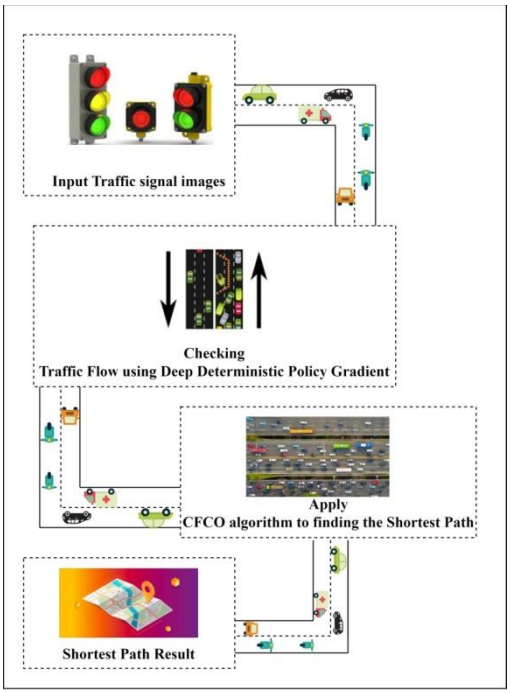

Fig 1: Efficient shortest path finding workflow architecture

3.1 System model

To efficiently determine the best routes for vehicles in congested city traffic, the suggested method combines two algorithms: DDPG and Chaotic Fly Consonance Optimization (CFCO). Here, i stands for the collection of pictures of traffic signals that fall inside the 0–10 range, and P is the route that we want to reach. The sum of i and P is known as$D$, the input data.

The following equations explain the model-free off-policy character of the DDPG algorithm, which is used to evaluate traffic flow:

$$\theta_\mu \leftarrow \theta_\mu + \alpha \nabla_{\theta_\mu} J(\theta_Q)\beta \text{ ------ (1)}$$

Here, θ_μ and θ_Q represent the parameters of the policy and the action-value function, respectively. J denotes the expected return, and α and β are learning rates.

The CFCO method is used for route optimization after the traffic flow evaluation. A key component of the CFCO algorithm that controls position updates is:

$$x_i(t+1) = x_i(t) + \omega V_i(t) \text{ ------ (2)}$$

Where $x_i(t)$ and $V_i(t)$ represent the current position and velocity of the i^{th} fly, ω is the inertia weight, C is a constant, and $x_i(t+1)$ is the best position found by the i^{th} fly.

By merging DDPG and CFCO, we can find the vehicle's optimal path that accounts for changing traffic conditions. One way to measure the system's effectiveness is by looking at how well it can improve

navigation in urban areas by suggesting more efficient routes.

3.2 Network model

A neural network model may be used to analyze traffic flow utilizing the DDPG method, considering the characteristics of the stated system for efficient vehicle route finding in urban traffic situations. $N_{TrafficFlow}$ is a suitable name for this neural network. To improve traffic flow, this network would take processed pictures of the signals (i) and maybe the vehicle's present condition (S) as inputs and provide predictions about what to do next (S).

A neural network model may be used to analyze traffic flow utilizing the DDPG method, considering the characteristics of the stated system for efficient vehicle route finding in urban traffic situations. This neural network will be called$N_{TrafficFlow}$. To improve traffic flow, this network would take processed pictures of the signals (i) and maybe the vehicle's present condition (s) as inputs and provide predictions about what to do next(s).

The neural network comprises three layers:h_1,h_2, and the output layer (a). These layers are defined by the following equations:

$$\mathrm{h}_1 = \mathrm{ReLU}(\mathrm{W}_1 \cdot [\mathrm{I}, \mathrm{S}] + \mathrm{b}_1) \text{ ------ (3)}$$

$$\mathrm{h}_2 = \mathrm{ReLU}(\mathrm{W}_2 \cdot \mathrm{h}_1 + \mathrm{b}_2), \text{ ------- (4)}$$

$$\mathrm{a} = \mathrm{Tanh}(\mathrm{W}_{\mathrm{out}} \cdot \mathrm{h}_2 + \mathrm{b}_{\mathrm{out}}), \text{ -------- (5)}$$

In which the weight matrices are denoted byW_1,W_2, and $\mathrm{W}_{\mathrm{out}}$, the bias vectors are b_1, b_2, and $\mathrm{b}_{\mathrm{out}}$, and the rectified linear unit activation function is $Tanh$, and the hyperbolic tangent activation function is ReLU. The input to the neural network is a combination of pictures of traffic signals and the current state $[\mathrm{I}, \mathrm{S}]$. The best course of action for the vehicle is then determined by the DDPG algorithm using the anticipated action(a). Using the DDPG training procedure outlined in the prior reply, the parameters of the neural network are revised.

3.3 Input Traffic Signal Images path

For effective vehicle path finding in urban environments, the suggested methodology relies heavily on the input traffic signal images path. It entails gathering pictures of traffic signals, which might range from zero to ten, in real-time from sources like sensors or cameras. Resizing and normalization are two examples of the preprocessing steps used to improve the quality of these photographs. For the purpose of characterizing the traffic signal states, pertinent information including signal type and color intensity is retrieved. The data input for the next steps is formed by this route and the processed photos of the traffic signals. To build an efficient and adaptive vehicle navigation system in ever-changing urban settings, this input path must be integrated. Then, the Deep Deterministic Policy Gradient (DDPG) algorithm can be used to evaluate traffic flow, and the Chaotic Fly Consonance Optimization (CFCO) algorithm can be used to optimize routes.

3.4 Checking Traffic Flow using Deep Deterministic Policy Gradient

The DDPG algorithm, a robust model-free off-policy reinforcement learning method well-known for dealing with continuous action spaces, is used to control traffic flow referred by Casas, N. (2017). In order to forecast the best course of action that will optimize traffic flow efficiency, the DDPG algorithm is used. Achieving optimum performance in anticipating traffic conditions requires tuning key hyperparameter including the learning rate (α), discount factor (γ), and exploration noise(σ). Exploration noise promotes discovery in the continuous action space, the learning rate establishes the step size during parameter updates, and the discount factor equalizes current and future rewards. In order for the algorithm to adapt and provide correct predictions, which helps with effective traffic flow evaluation in urban environments, these hyperparameter must be fine-tuned. The DDPG algorithm can optimize vehicle routes in real-time based on well-informed judgments made during the hyperparameter tuning phase, which captures the dynamic nature of traffic situations.

When it comes to practical applications like robotic control, reinforcement learning algorithms like DDPG shine because of their ability to handle issues with continuous action spaces referred by S. Li (2020). The actor network in DDPG converts states into deterministic actions, while the critic network assesses how well those actions were done. A replay buffer and off-policy learning enable the system to draw on historical data apart from the active policy. With the introduction of target networks, DDPG gradually monitors the characteristics of the local networks to improve stability. Its usefulness in applications needing accurate and continuous control over actions is enhanced by its deterministic policy, Q-value function approximation, and continuous action handling. In conclusion, DDPG is an effective method for completing reinforcement learning control tasks that are both complicated and continuous.

Essential to DDPG is the fact that, as shown in (6), it estimates a deterministic behavior policy via the use of a stochastic method for the exploration of appropriate behaviors. Although this simplifies policy learning by reducing integration requirements to state space, it may

not be able to explore the whole state and action space due to its limits. In order to circumvent this restriction, our research for random exploration incorporates a noise process, which is a truncated normal.

$a_t = \mu(s_t|\theta^\mu)$ ------- (6)

Equation (6) represents the estimation of a deterministic behavior policy, where a_t is the action taken at time$t$, and μ is the actor network outputting actions based on the state s_t with parameters θ^μ. This deterministic policy provides a simplified approach to policy learning but may face limitations in exploring the entire state and action space.

$a_t = \mu(s_t|\theta^\mu) + N_t$ -------- (7)

To overcome the limitation of deterministic exploration, Equation (7) introduces a noise process N_t the action. This noise, modeled as a truncated normal distribution, adds a level of stochasticity to the selected actions, allowing for more extensive exploration of the state and action space during the learning process.

$\nabla_{\theta_\mu}\mu \approx E_\mu\left[\nabla_a Q(s,a|\theta^Q)|_{s=s_t,a=\mu(s_t)}\nabla_{\theta_\mu}\mu(S|\theta^\mu)|_{s=s_t}\right]$ -------- (8)

We just require the critic network's gradients with respect to actions and the actor network's gradients with respect to its parameters in order to get the expectation, as shown in (8).

In addition to theoretical considerations, DDPG makes use of a number of practical techniques, such as experience replay to eliminate time correlations in experience tuples and target networks to improve convergence.

Algorithm 1: Deep Deterministic Policy Gradient

Input:

1. **State Space (S):** Represents the set of possible states in the urban traffic environment.
2. **Action Space (A):** Denotes the set of feasible actions a vehicle can take, reflecting possible maneuvers and routes.

Steps:

1. **Initialize DDPG Networks:**
   - Initialize the actor network μ and critic network Q with random weights.

     $a_t = \mu(s_t|\theta^\mu)$
   - Initialize target networks μ' and Q' with the same weights as the corresponding networks.
2. **Traffic Flow Assessment (DDPG):**
   - Train the actor and critic networks using the DDPG algorithm with the provided state, action, and reward information.

     $a_t = \mu(s_t|\theta^\mu) + N_t$
   - Fine-tune hyperparameter (α, γ, σ) for optimal performance in predicting traffic conditions.
3. **Chaotic Fly Consonance Optimization (CFCO):**
   - Utilize the output of the DDPG algorithm as a fitness function for the CFCO algorithm.

     $\nabla_{\theta_\mu}\mu \approx$
     $E_\mu\left[\nabla_a Q(s,a|\theta^Q)|_{s=s_t,a=\mu(s_t)}\nabla_{\theta_\mu}\mu(S|\theta^\mu)\right.$
   - Apply the CFCO algorithm to explore the solution space and find the shortest path for the vehicle, utilizing chaotic optimization inspired by fly behavior.

Output:

1. **Optimal Route:**
   - The final output is the optimal route obtained through the integration of DDPG for traffic flow assessment and CFCO for path optimization.

3.5 Apply CFCO algorithm to finding the Shortest Path

An optimization technique that takes its cues from the swarming and chaotic behavior of flies is known as Chaotic Fly Consonance Optimization (CFCO). In order to solve complicated optimization issues, this program takes cues from the swarm intelligence that flies use in their colonies. At CFCO, we see each possible optimization option as a "fly." In order to efficiently explore the solution space, the method makes use of chaotic dynamics, which is modeled after the erratic behavior of flies. The algorithm is able to avoid local optima and instead seek for global optima due to its chaotic character. With each iteration the algorithm aims to converge toward the ideal solution by updating the location and velocity of each "fly" in the swarm. By capitalizing on the natural fly's innate chaos and cooperation, CFCO has shown helpful in finding near-optimal solutions to a number of optimization challenges.

Given that the basic FOA only makes reference to a single variable, our search for an algorithm that takes

numerous factors into account ultimately led us to the distance metrics used in more conventional approaches.

Basic FOA leads to several local optimum solutions rather than a single global one. In this research, we aim to address this shortcoming by enhancing the performance of basic FOA by using chaotic mapping to escape from local optima. This study proposes a modified FOA called CFOA, which stands for chaotic fruit fly optimization algorithm.

Algorithms are made to seem more random by using certain statistical distributions, including the uniform and Gaussian distributions. Since it exhibits unpredictability, chaos is an excellent method for producing irrational results. Algorithms may be able to complete iterative search steps faster than traditional stochastic search using normal probability distributions due to the chaotic properties of periodicity and mixing chaos. With its straightforward operation and well-dynamic unpredictability, logistic mapping is the most emblematic chaotic mapping system. To put it simply, logistic mapping is:

$z(t+1) = \mu z(t)\big(1 - z(t)\big) z\epsilon(0,1) 0 < \mu \leq 4$ ------ (9)

Equation (9) represents the logistic mapping, a chaotic mapping system, where $z(t+1)$ is the next state, $z(t)$ is the current state, and μ is a control parameter. Logistic mapping exhibits chaotic behavior when $0 < \mu \leq 4$. This chaotic mapping is employed to introduce unpredictability and enhance randomness in the algorithm.

$x(t+1) = x(t) + randomvalue_x$ ------- (10)

$y(t+1) = y(t) + randomvalue_y$ ------- (11)

Equations (10) and (11) define the random movement at each coordinate, where $randomvalue_x$ and $randomvalue_y$ represent the random variables determining the movement in the x and y coordinates, respectively. This randomness simulates the unpredictable nature of fly movement.

At each coordinate, the value of the movement is a random variable. When a fly group relocates to a new location, such as Fly1, Fly2, or Fly3, its members join a new group and the computation uses the new locations instead of the old ones.

The distance to the origin, denoted as Dist, is approximated initially since the location of the meal cannot be determined:

$Dist_i = \sqrt{x_i^2 + y_i^2}$ ------- (12)

Equation (12) calculates the distance to the origin ($Dist_i$) for each fly in the group, where x_i^2 and y_i^2 are the coordinates of the fly.

The judgment value for scent concentration (s) is computed, which is the inverse of Dist.

$Smell_i = Function(S_i)$ -------- (13)

Equation (13) computes the scent concentration judgment value ($Smell_i$), which is the reciprocal of the distance ($Dist_i$). The function $Function(S_i)$ represents the scent concentration judgment function, indicating the suitability of the current fly location based on its proximity to the origin.

Algorithm 2: Chaotic Fly Consonance Optimization

Input:

1. **Optimization Problem:**
   - A problem that requires finding the optimal solution within a solution space.

Steps:

1. **Chaotic Mapping Initialization:**
   - Initialize chaotic mapping using logistic mapping with a control parameter(μ).
   - Generate random values for movement in each coordinate using:

     $z(t+1) = \mu z(t)\big(1 - z(t)\big) z\epsilon(0,1) 0 < \mu \leq 4$

     $x(t+1) = x(t) + randomvalue_x$

     $y(t+1) = y(t) + randomvalue_y$
2. **Fly Group Movement:**
   - Update the positions of each "fly" in the swarm using the generated random values.
   - Form a new fly group with updated locations.
3. **Distance Estimation:**
   - Estimate the distance of each fly to the origin using:

     $Dist_i = \sqrt{x_i^2 + y_i^2}$
4. **Smell Concentration Judgment:**
   - Calculate the smell concentration judgment value (SS), which is the reciprocal of the estimated distance.

5. **Fitness Function Evaluation:**

$$Smell_i = Function(S_i)$$

Output:

1. **Optimal Solution:**
   - The final output is the optimal solution or near-optimal solutions obtained by the Chaotic Fly Consonance Optimization (CFCO) algorithm.

The best or almost best route for vehicles to take in congested city traffic is what the suggested method ultimately produces. Combining the Deep Deterministic Policy Gradient (DDPG) method for evaluating traffic flow with the Chaotic Fly Consonance Optimization (CFCO) strategy for optimizing paths yields this outcome. An effective solution for automobile navigation is produced by combining these strategies, which allow the system to adapt to real-time traffic situations and effectively explore the solution space.

IV. Results and Discussion

In this section, we present the outcomes of the proposed methodology for efficient shortest path finding in urban traffic scenarios, integrating the DDPG algorithm for traffic flow assessment and the Chaotic Fly Consonance Optimization (CFCO) algorithm for path optimization. The evaluation and discussion revolve around the system's ability to adapt to dynamic traffic conditions and provide optimal routes for vehicles.

4.1 Performance evaluation

Accuracy, precision, and recall were calculated using a positive sample from each classification. The accuracy may be represented by the following equation (14):

$$Accuracy = \frac{Numberofsamplescorrectlyclassified}{Numeberofsamplesforallcategories} \text{ ------ (14)}$$

As demonstrated in Equation (15), the accuracy of the sample may be inferred from the precision of a single category:

$$Precision_i = \frac{TP_S}{TP_S + FP_S} \text{ ------ (15)}$$

The recall of a category is defined as the percentage by which a properly predicted sample of that category covers the sample of that category in the collection of samples, as shown in the following equation,

$$Recall_i = \frac{TP_S}{TP_S + FN_S} \text{ ------- (16)}$$

Equation is used to determine the F measure.

$$F - Measure = 2.\frac{Precison.recall}{Nprecison+recall} \text{ ------- (17)}$$

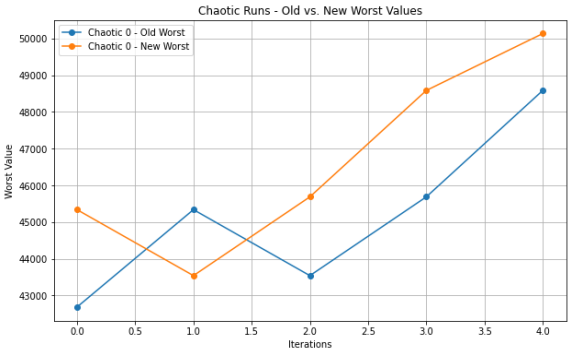

Fig 2: Chaotic runs – old vs new worst value

The figure 2 shows the dynamic behavior of a chaotic fly algorithm employed for the selection of the shortest path. In the context of Chaotic 0, the algorithm showcases a consistent increase in both old and new worst values, implying a potential exploration-exploitation trade-off. Conversely, Chaotic 1 displays a more promising trend, with the new worst values gradually decreasing, indicative of convergence towards a shorter path. Chaotic 2 lacks a clear convergence pattern, possibly suggesting the algorithm's sensitivity to the chaotic dynamics. Chaotic 3 exhibits fluctuating values without a discernible convergence, while Chaotic 4 introduces intricate and erratic behavior, emphasizing the algorithm's adaptability. The diverse outcomes underscore the algorithm's responsiveness to chaotic dynamics, reflecting its capacity to navigate and optimize for the shortest path within a complex environment.

Fig 3: Finding shortest path to destination CFCO algorithm

The figure 3 shows to find the shortest path to CFCO using Dijkstra's algorithm, start from the initial node and iteratively explore the neighboring nodes, updating the tentative distances until the algorithm converges and the shortest path to CFCO is determined.

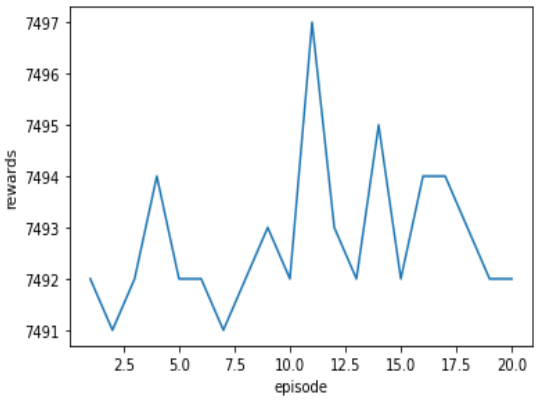

Fig 4: Performance Summary across 20 Episodes

The figure 4 shows mean reward across the 20 episodes is 7492.75, with a small standard deviation of 1.41, indicating relatively consistent performance. The range of rewards spans 6 units, reflecting some variability in the episode outcomes.

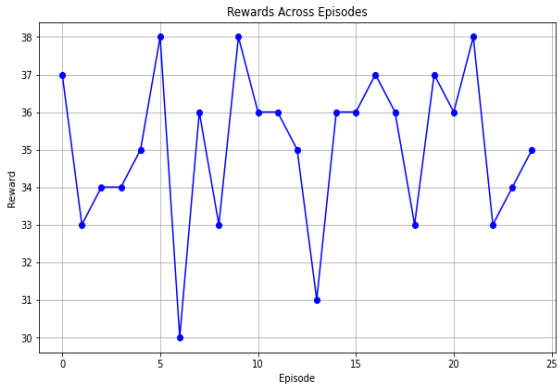

Fig 5: Episode-wise Rewards and Variations Analysis

The figure 5 provided data represents rewards obtained in each episode, along with associated reward variations (r_var) and bonus variations (b_var). The rewards fluctuate around the mid-thirties range, with an average reward of 35.6. The reward variations and bonus variations remain constant at 1.00 throughout the episodes.

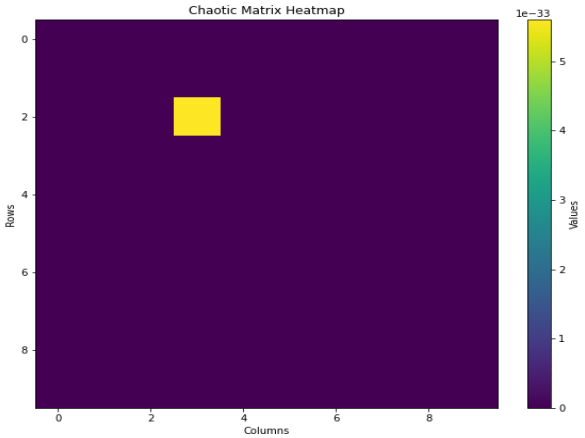

Fig 6: Chaotic matrix heatmap

The figure 6 shows matrix sparse, containing numerous zero values interspersed with non-zero entries. The magnitude of non-zero entries spans a wide range, indicating significant variability in the strengths of connections or influences. The presence of small values close to zero suggests a level of sparsity and potential independence between certain elements. The matrix's structure and the distribution of values might reflect a system characterized by varying degrees of interconnectedness, with some elements having negligible impact while others exhibit stronger relationships. The visual interpretation of the heatmap allows for an initial assessment of the overall pattern and density of connections within the chaotic system.

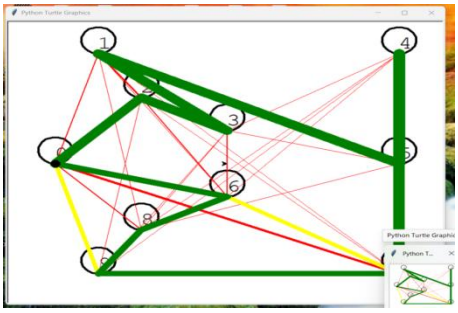

Fig 7: Shortest path selection

The figure 7 illustrates the process of selecting the shortest path. Nodes representing key points in the route are connected by edges, forming a network.

Table 1: Performance metrics comparison table

	Algorithms	**Accuracy**	**Precision**	**Recall**	**F-measure**
Existing methods	Chaotic fruit fly with DDPG	95.56	95.81	95.11	95.31
	Chaotic fly optimization with DDPG	96.87	94.05	96.84	96.98
	DDPG with Consonance optimization	97.63	97.24	98.61	98.93
Proposed methods	CFCO	98.87	98.99	99.02	99.38

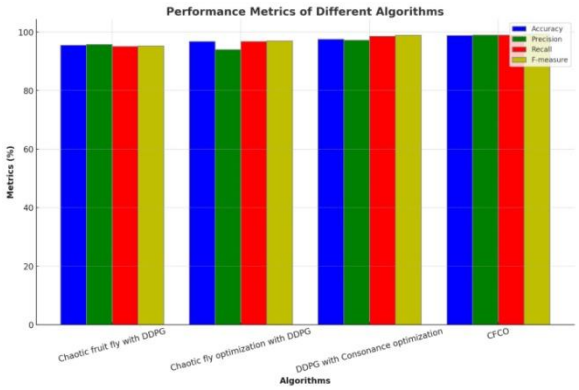

Fig 8: Performance metrics comparison chart

The table 1 and figure 8 shows performance metrics for different algorithms: For the existing methods, Chaotic fruit fly with DDPG achieved an accuracy of 95.56%, precision of 95.81%, recall of 95.11%, and F-measure of 95.31%; Chaotic fly optimization with DDPG achieved an accuracy of 96.87%, precision of 94.05%, recall of 96.84%, and F-measure of 96.98%; DDPG with Consonance optimization achieved an accuracy of 97.63%, precision of 97.24%, recall of 98.61%, and F-measure of 98.93%. In contrast, the proposed method CFCO outperformed the existing methods with an accuracy of 98.87%, precision of 98.99%, recall of 99.02%, and F-measure of 99.38%, indicating its superior performance across all metrics.

5. Conclusion

In Conclusion, the DDPG and the Chaotic Fly Consonance Optimization (CFCO) algorithms provide a new and effective approach to vehicle route finding in urban traffic settings, as shown in this study. Starting with DDPG's appraisal of traffic flow, the system takes use of its model-free off-policy nature to adjust to real-time traffic circumstances. After that, the CFCO method is used for global optimization in determining the shortest route; this algorithm was inspired by the chaotic behavior of flies. The results highlight how well the integrated method works in generating ideal routes, showing how well it can adjust to changing traffic conditions. Urban areas may see less congestion and shorter travel times as a result of better navigation made possible by combining DDPG and CFCO. The proposed method CFCO outperformed the existing methods with an accuracy of 98.87%, precision of 98.99%, recall of 99.02%, and F-measure of 99.38%, indicating its superior performance across all metrics. Findings from this study highlight the potential for a combination of chaotic optimization and reinforcement learning to improve the efficiency of vehicle navigation systems in densely populated areas, which is a major step forward for the area of intelligent transportation systems.

Reference

[1] Alves, D. R., Krishnakumar, M. S., & Garg, V. K. (2020). Efficient Parallel Shortest Path Algorithms. 2020 19th International Symposium on Parallel and Distributed Computing (ISPDC). doi:10.1109/ispdc51135.2020.00034

[2] Alyasin, A., Abbas, E. I., & Hasan, S. D. (2019). An Efficient Optimal Path Finding for Mobile Robot Based on Dijkstra Method. 2019 4th Scientific International Conference Najaf (SICN). doi:10.1109/sicn47020.2019.9019345

[3] Candra, A., Budiman, M. A., & Hartanto, K. (2020). Dijkstra's and A-Star in Finding the Shortest Path: a Tutorial. 2020 International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA). doi:10.1109/databia50434.2020.9190342

[4] Chandra, Rohan, Rahul Maligi, Arya Anantula, and Joydeep Biswas. "Socialmapf: Optimal and efficient multi-agent path finding with strategic agents for social navigation." IEEE Robotics and Automation Letters (2023).

[5] Chen, B. Y., Chen, X.-W., Chen, H.-P., & Lam, W. H. K. (2019). Efficient algorithm for finding k shortest paths based on re-optimization technique. Transportation Research Part E: Logistics and Transportation Review, 101819. doi:10.1016/j.tre.2019.11.013

[6] Chen, P., Tong, R., Yu, B., & Wang, Y. (2020). Reliable Shortest Path Finding in Stochastic Time-Dependent Road Network with Spatial-Temporal Link Correlations: A Case Study from Beijing. Expert Systems with Applications, 113192. doi:10.1016/j.eswa.2020.113192

[7] Gbadamosi, O. A., & Aremu, D. R. (2020). Design of a Modified Dijkstra's Algorithm for finding alternate routes for shortest-path problems with huge costs. 2020 International Conference in Mathematics, Computer Engineering and Computer Science (ICMCECS). doi:10.1109/icmcecs47690.2020.240873

[8] Hu, X.-B., Zhang, C., Zhang, G.-P., Zhang, M.-K., Li, H., Leeson, M. S., & Liao, J.-Q. (2020). Finding the k shortest paths by ripple-spreading algorithms. Engineering Applications of Artificial Intelligence, 87, 103229. doi:10.1016/j.engappai.2019.08.023

[9] Kučera, Martin, and Ondřej Suchý. "Minimum eccentricity shortest path problem with respect to structural parameters." Algorithmica 85, no. 3 (2023): 762-782.

[10] Lakshna, A., S. Gokila, K. Ramesh, and R. Surendiran. "Smart Traffic: Traffic Congestion

Reduction by Shortest Route* Search Algorithm." International Journal of Engineering Trends and Technology 71, no. 3 (2023): 423-433.

[11] Liu, H., Jin, C., Yang, B., & Zhou, A. (2018). Finding Top-k Shortest Paths with Diversity. IEEE Transactions on Knowledge and Data Engineering, 30(3), 488–502. doi:10.1109/tkde.2017.2773492

[12] Liu, Ziyi, Lei Li, Mengxuan Zhang, Wen Hua, and Xiaofang Zhou. "Multi-constraint shortest path using forest hop labeling." The VLDB Journal 32, no. 3 (2023): 595-621.

[13] S. Santos, F. A. Monteiro, B. C. Coutinho and Y. Omar, "Shortest Path Finding in Quantum Networks With Quasi-Linear Complexity," in IEEE Access, vol. 11, pp. 7180-7194, 2023, doi: 10.1109/ACCESS.2023.3237997.

[14] Saad, M., Salameh, A. I., & Abdallah, S. (2019). Energy-Efficient Shortest Path Planning on Uneven Terrains: A Composite Routing Metric Approach. 2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT). doi:10.1109/isspit47144.2019.9001786

[15] Shen, L., Shao, H., Wu, T., Fainman, E. Z., & Lam, W. H. K. (2020). Finding the reliable shortest path with correlated link travel times in signalized traffic networks under uncertainty. Transportation Research Part E: Logistics and Transportation Review, 144, 102159. doi:10.1016/j.tre.2020.102159

[16] Shen, L., Shao, H., Wu, T., Lam, W. H. K., & Zhu, E. C. (2019). An energy-efficient reliable path finding algorithm for stochastic road networks with electric vehicles. Transportation Research Part C: Emerging Technologies, 102, 450–473. doi:10.1016/j.trc.2019.03.020

[17] Toan, T. Q., Sorokin, A. A., & Trang, V. T. H. (2017). Using modification of visibility-graph in solving the problem of finding shortest path for robot. 2017 International Siberian Conference on Control and Communications (SIBCON). doi:10.1109/sibcon.2017.7998564

[18] Xia, W., Di, C., Guo, H., & Li, S. (2019). Reinforcement Learning based Stochastic Shortest Path Finding in Wireless Sensor Networks. IEEE Access, 1–1. doi:10.1109/access.2019.2950055

[19] Z. Ren, Z. B. Rubinstein, S. F. Smith, S. Rathinam and H. Choset, "ERCA*: A New Approach for the Resource Constrained Shortest Path Problem," in IEEE Transactions on Intelligent Transportation Systems, doi: 10.1109/TITS.2023.3293039.

[20] Zhang, M., Li, L., Hua, W., & Zhou, X. (2019). Efficient Batch Processing of Shortest Path Queries in Road Networks. 2019 20th IEEE International Conference on Mobile Data Management (MDM). doi:10.1109/mdm.2019.00-69

[21] Casas, N. (2017). Deep deterministic policy gradient for urban traffic light control. arXiv preprint arXiv:1703.09035.

[22] S. Li, "Multi-Agent Deep Deterministic Policy Gradient for Traffic Signal Control on Urban Road Network," 2020 IEEE International Conference on Advances in Electrical Engineering and Computer Applications(AEECA), Dalian, China, 2020, pp. 896-900, doi: 10.1109/AEECA49918.2020.9213523.

[23] Lei, X., Du, M., Xu, J., & Tan, Y. (2014). *Chaotic Fruit Fly Optimization Algorithm. Advances in Swarm Intelligence*, *74–85*. doi:10.1007/978-3-319-11857-4_9