Intelligent Car Park Assist Using Fish Swarm Algorithm

SPYRIDON A. KAZARLIS,
Dept. of Informatics, Computers and Telecommunications Engineering
International Hellenic University,
Terma Magnesias St., 62124 Serres,
GREECE

Abstract:—In this paper a novel Intelligent Car Park Assist System is presented that uses an Evolutionary Computation method called "Fish Swarm Algorithm" or FSA, inspired from the behavior of schools of fish. The work focuses on the "parallel park" problem where a car needs to park between two already parked cars with a gap between them. The modeled car adopts the Ackermann model for moving, and has to complete the park task in 3 moves. Every move consists of steering the wheel at an angle and moving the car at a specific distance. For finding the optimal car moves to park successfully, a paradigm of the FSA algorithm is used. Simulation results confirm both the modelling soundness and the optimization capabilities of the FSA method.

Key-Words: - Fish Swarm Algorithm, Intelligent Car Park Assist, Evolutionary Computation, Ackermann Steering Model.

Received: March 29, 2024. Revised: September 11, 2024. Accepted: October 13, 2024. Published: December 4, 2024.

1. Introduction

This paper presents the application of an Evolutionary Computation method called "Fish Swarm Algorithm" or FSA on the problem of parallel car park in an automated and optimized way, leading to an Intelligent Car Park Assist system.

In this section, a brief introduction on both the "Parallel Car Park problem" and the "Fish Swarm Algorithm" will be given.

1.1 Parallel Car Park Problem

The parallel car park task [1] is one of the most difficult to master driving maneuvers, especially for young drivers.

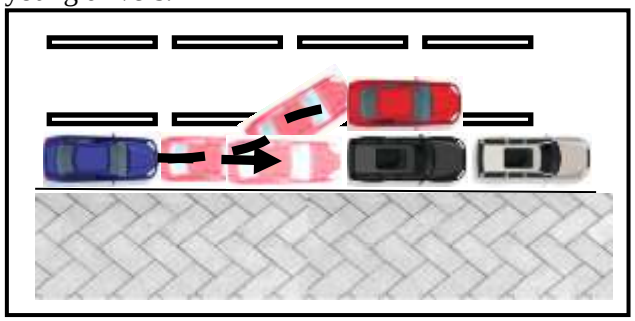

Fig.1 Depiction of the parallel car park problem

And in most countries it is a mandatory task to accomplish in the practical exams for getting a driving license. For this reason several car manufacturers offer "Intelligent Parallel Park Assist" systems, especially in their high priced models, in order to assist the driver in such situations. Cars equipped with such systems offer a wide range of help for the driver, from simply indicating the distance of the car to the front and back cars, to showing him an upper 360 degrees view of the car and its surroundings, or finally perform the parallel park completely autonomously, with the driver holding the role of a passenger.

Historically, such systems date back to 1990, where the French Institute for Research in Computer Science (INRIA) developed an autonomous car park system for a Ligier car [2]. In more recent years (2003) Toyota Motor Corporation introduced the Intelligent Parking Assist System (IPAS) that was first placed on a Toyota Prius hybrid model [3]. Later, such systems were placed on Lexus cars (2006), the Ford Lincoln model by Ford in 2009 as "Active Park Assist" system [4], and the BMW series 5 model in 2010, as the 'Parking Assistance" system. Following that, in 2012 Ford generalized the use of its "Active Park Assist" on more models, like the Ford Focus, Fusion, Escape, Explorer and Flex. The same year Toyota produced an upgraded system that equipped models like the Toyota Prius 5, Lexus LS460 and LS460L. Audi introduced a similar system placed on the A6 model, and Mercedes Benz developed the "Parktronic" system

that equipped models like CLS-Class, C-Class, E-Class, S-Class, M-Class, etc. In 2014 Jeep Company supplied its Cherokee model with its "Parksense" system and in 2015 BMW put a similar system on the BMW i3 model that was controlled by a smartwatch.

In the literature, there are many scientific papers contributing to the solution of this problem, like Xu, Chen and Xie [5], who propose a system based on vision that is able to recognize a possible parking place and find a path for driving the car in it. In the work of Wang and Zhu [6] a fuzzy controller is constructed for an intelligent car park system. In the work of Ji, Wang, Zhao, Lin, Zang & Li [7] a neural network is trained to act as a PID controller for vehicle parallel parking. In the work of Saleh, Ismail & Riman, [8] a new geometric-based algorithm is proposed for parallel car parking of a car-like robot that achieves better performance, especially in narrow spaces. In the work of Nakrani & Joshi [9], the authors propose a Fuzzy system to cope with parallel parking in dynamically changing environments and they continue their research in [31] where they propose a fuzzy-based adaptive dimension parking algorithm (FADPA) that also copes for obstacle avoidance. And in the work of Rashid, Rahman, Islam, Alwahedy & Abdulahi [10] the authors propose yet another fuzzy logic controller for automated parallel parking.

1.2 Fish Swarm Algorithm

Fish Swarm Algorithm (FSA or AFSA) is an Evolutionary Computation optimization method, first introduced by Li et al. in 2002 [11], and belongs to the family of Swarm Intelligence methods [12]. It is inspired by the behavior of schools of fish in their quest for prey. It employs artificial fish in the role of agents that search the search space of an optimization problem. It mimics four different behaviors found in schooling fish, namely: Prey, Swarm, Follow and Random behavior.

Prey behavior simulates the individual quest of every fish to find prey or food (good solutions) within its visibility distance (Visual). Swarm behavior mimics the fish clustering tendency, where every fish tends to avoid danger by assembling in a group (convergence). Follow behavior simulates the behavior of an individual fish to follow neighbor fish that have already found food sources (exploit good solutions). Finally, Random behavior gives every fish (or agent) the chance to search the surrounding space for food (good solutions) independently of the other fish, and in a random manner. In figure 2 a simplified pseudocode of the FSA method is provided, where Visual is a metric of the radius inside which the fish can see its surrounding fish, and Step is the maximum distance the fish can travel when its position changes.

```
Read N, Visual, Step, Try, Iter; i=0
For fish=1 to N
 position(fish) = random
 fitness(fish) = Fitness (position(fish))
Endfor
While (i < Iter)
  For fish=1 To N
    calculate no of neighbors nf of fish (inside Visual)
    If (nf >= 1) then     // Follow Behavior
      find neigboring fish nfb with best fitness
      If (fitness(nfb) > fitness(fish) then
          Move fish a Step towards nfb
      else      //Swarm behavior)
          calculate the geometrical center nfc of neighbor fish
          If (fitness (nfc) > fitness(fish)) then
            Move fish a Step towards nfc
          Else          //Prey behavior)
            t=0
            While (t<Try AND Better Solution Not Found)
              select a random position rfv inside Visual
              If (fitness(rfv)>fitness(fish)) then
                    move fish a Step towards rfv
              Endif
              t = t + 1
            Endwhile
            If (Better Solution Not Found) then
              //Random behavior
              move fish a random Step inside Visual
            Endif
          Endif
      Endif
    Else        //Prey behavior)
      t=0
      While (t<Try AND Better Solution Not Found)
          select a random position rfv inside Visual
          If (fitness(rfv)>fitness(fish)) then
            move fish a Step towards rfv
          Endif
          t = t + 1
      Endwhile
      If (Better Solution Not Found) then
          //Random behavior
          move fish a random Step inside Visual
      Endif
    Endif
  Endfor
  i = i + 1;
Endwhile
```

Fig.2 Pseudocode of the FSA optimization method

Since 2002, there have been many papers in the literature concerning improved and hybrid models of FSA to solve different real-world optimization problems.

Such problems include data clustering [13], image segmentation [14], fault diagnosis [15], power allocation scheme [16], parameter optimization of the deep auto-encoder [17], spoiled meat detection [18], manufacturing [19], risk probability prediction [20], guidance error estimation [21], navigation [22], wireless sensor network [23], energy management [24] and motion estimation in video encoding [30].

1.3 Paper Structure

In Section 2 a thorough description is given for the exact parallel car park problem that is dealt with in this work. In Section 3 the Fish Swarm Algorithm used in this work is discussed. Section 6 describes the parameters and the setup for the simulation experiments, while Section 5 presents the results of the executed simulation experiments. Finally conclusions are discussed in Section 6.

2. The Car Park Problem

In this work a simple version of the Parallel Car Park problem [1], [6], [7], [8], [9], [10] is adopted, that can be seen in figure 3.

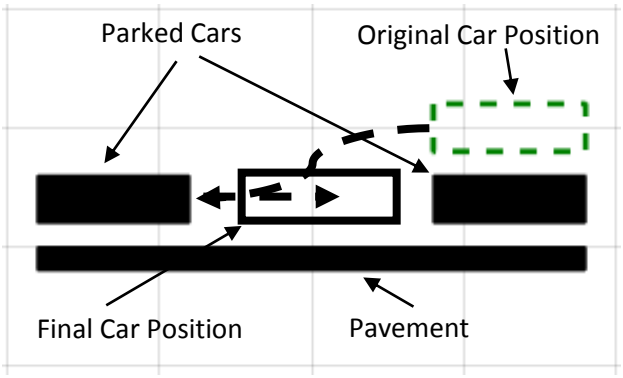

Fig.3 Depiction of the Parallel Car Park problem.

The type of car employed in this work has the following characteristics:

Length:	5 meters
Wheelbase:	3 meters
Width:	2 meters
Steering Model:	Ackermann
Steering Angle:	±30 degrees

The car movement is modeled after the Ackermann model [25], [26] that can be described by the differential equations shown in eq.1.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 \\ \sin(\theta) & 0 \\ \tan(\psi)/l & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \dot{\psi} \end{bmatrix} \quad \text{Eq.1}$$

where x, y denote the global vehicle position in meters, l (el) is the wheelbase in meters, θ is the global vehicle heading in radians, ψ is the vehicle steering angle in radians, and v is the vehicle speed in m/sec.

The gap between the parked cars is set to 8 meters and the initial car position is set as for the car to be parallel to the second parked car (Fig.3) and having a one (1) meter clearance from it.

The car movement is encoded in the following way: the car is allowed to make three (3) distinct movements in order to park successfully. Each movement is described by a) the steering angle, and b) the travel distance of the car, as shown in eq.2

$$\text{Movement}_i, (i=1..3) = [\ \text{Steering}_i\ ,\ \text{Distance}_i\ ] \quad \text{Eq.2}$$

The travel distance of the car is metered using the center of the rear axle as the reference travelling point. During a single movement, the steering angle remains constant and does not change. In this way, the whole movement can be described by a total of six (6) parameters, as shown in eq.3

$$\text{Solution}=\begin{bmatrix} Steering1 & Distance1 \\ Steering2 & Distance2 \\ Steering3 & Distance3 \end{bmatrix} \quad \text{Eq.3}$$

During the three movements, the simulation monitors and counts possible collisions with the parked cars or the pavement (Fig.3). Moreover, in the case of a collision, the simulation calculates how severe the collision was, by calculating the volume of the corresponding polygons of the car and obstacle that intersect. This collision metric, that is analog to the collision severity, is used to penalize the corresponding solutions, in a proportional way. This leads the FSA algorithm a) to avoid such "colliding" invalid solutions and b) it produces a gradient towards "no-collision" acceptable solutions that facilitates FSA to follow this gradient towards feasible optimality and converge to a final acceptable solution.

The objective function of each solution is judged according to the following criteria:

- The distance between the vehicle and the pavement
- The degree of parallelization of the car to the pavement
- The equality of distances of the car with the front and back obstacles (parked cars) or the degree of centering inside the empty space.

From the above, it is evident that this problem can be classified as a constrained multi-objective problem. Thus, in order to build a consolidated fitness function, one can build it as a weighted sum of the individual objective functions [27].

The objective value of each produced solution (sol) is calculated according to eq. 4.

$$\text{Obj(sol)}=a\cdot\text{pdist(sol)}+b\cdot\text{paral(sol)}+c\cdot\text{cent(sol)} \quad \text{Eq.4}$$

where pdist(sol) is the minimum distance between the car and the pavement, paral(sol) is the absolute difference of the y values of the centers of the front and rear axles, and cent(sol) is the absolute difference between the minimum distances of the car with the front and rear obstacles (parked cars). Coefficients a, b and c are weight coefficients that can be chosen arbitrarily, in order to emphasize in each optimization objective.

And in order to handle the collision constraints, one can build a penalty function for the violation of the constraints and add it to the objective function [28].

So, the fitness function of each produced solution is built by adding the penalty value to the objective value:

$$Fitness(sol) = Obj(sol) + p \cdot Penalty(sol) \quad \text{Eq.5}$$

where Penalty(sol) is a metric of possible collision of the car with the obstacles, that is proportional to the severity or the volume of the collision, and p is a penalty coefficient.

3. Fish Swarm Algorithm

The paradigm of FSA that is used in this work is more or less a classical FSA implementation and is based on the pseudocode of figure 2. A worth-mentioning detail that is omitted from the pseudocode, for simplicity reasons, is that in order to accept a new fish position (e.g. when in Follow behavior (fitness(nfb) > fitness(fish) is true) there is another extra criterion that has to be met. This criterion is the crowding criterion [29] that suggests that a new fish position will be accepted only if the new position is not very crowded with fish. In the literature this is expressed by the equation:

$$nf / n < \delta \quad \text{Eq.6}$$

where nf is the number of neighboring fish in the new position (within Visual), n is the total fish number and δ is a real number in the range [0..1]. This crowding criterion is added to the algorithm in order to avoid premature convergence of the algorithm and help it maintain a solution diversity, in order to facilitate and pluralize its search effort.
However, it is rather evident that in the early stages of FSA where the fish population will be scattered throughout the search space, eq. 6 will be easily satisfied. But as the population converges eq. 6 will be more and more difficult to satisfy as nearly all fish may probably end up within each-others visual range. To cope with this, one has to progressively adapt either the δ coefficient, or the Visual range of fish, during the FSA execution, in a gradient manner as the population of fish converges.
In order to circumvent this, this work introduces a different crowding criterion, which works in a relative and not absolute manner, and is shown in eq. 7

$$neighbors(newpos) / neighbors(oldpos) < d \quad \text{Eq.7}$$

where neighbors() is a function that returns the number of surrounding fish inside the Visual range, newpos is the new position under check to be occupied by the fish, and oldpos is the current fish position in the search space. Parameter d is a number a little greater than 1 (eg. d=1.2). This allows the fish to move to areas that are sparser than the current, but also allows them to move to areas that are also a little more crowded. So, in this way, the population is allowed to slowly converge towards the optimum. If d is less than one, then the population is prohibited from converging, as it prohibits the newpos from being more crowded than the oldpos. On the other hand, if d is much larger than 1 then FSA will converge at a high rate (premature convergence). So, a middle value of, let's say, 1.1 to 1.5 should be chosen.
In this work, all code has been developed and executed in the MATLAB R2020a environment.

4. Simulation Parameters

In order to conduct the simulations and obtain results, the following FSA parameters have been used:
The value of Visual has been set to 5 for the "Steering Angle" parameters and to 0.5 for "Travel Distance" parameters. The value of Step has been set to 60% of Visual. The number Try that denotes the number of tries to find prey, during the Prey behavior, has been set to 3. Crowding parameter d has been set equal to 1.5. Concerning the fitness function (Eq. 4 & 5) the parameter values are set as: a=2, b=2, c=1, p=10.
Concerning the Ackermann steering model, the number of allowed moves is 3, having 2 parameters for every move (Steering and Distance), leading thus to a total number of 6 variables (Eq. 3). The range of each Steering variable is [-30..30] degrees, and the range of each Travel-Distance variable is [-5..5] meters.
The simulation setup was twofold and includes Setup1 and Setup2. In Setup1 the fish number N is equal to 50 and the number of algorithm iterations "Iter" is set to 100. In Setup2 the number of fish was doubled and the parameter values were N=100 and Iter=100.

5. Simulation Results

The FSA algorithm is a stochastic algorithm. This dictates that a number of simulation runs have to be conducted in order to extract useful performance results.
For this reason a number of ten (10) runs have been executed for each setup (Setup1 and Setup2). The results can be shown in Table 1 and Table 2 respectively.
In the result tables, negative steering angles mean right turns, while positive steering angles mean left turns. Similarly, positive distances mean forward motion while negative ones mean moving in reverse. Each table line shows the best solution achieved at each run.
As can be seen from the results Setup1 achieves a

50% success rate, while Setup2, that uses double the number of fish, achieves an 80% success rate. In fact every fitness value below 2.5 corresponds to a successful parking. Two such successful but different cases, found by FSA, are depicted in figures 4 and 5.

Table 1. Results of the FSA for Setup1

Run	Steer. Angle 1	Steer. Angle 2	Steer. Angle 3	Distance 1	Distance 2	Distance 3	Fitness	Result
1	-29,98	-10,55	29,99	-4,22	-0,66	-3,60	3,303215	Fail
2	-14,01	11,21	-19,03	-4,84	-3,45	1,52	4,379467	Fail
3	-23,96	29,87	13,81	-4,61	-3,88	1,37	3,065357	Fail
4	-16,27	-29,81	29,95	1,28	-4,98	-3,73	2,12648	Succ
5	29,98	-15,48	26,48	1,80	-4,77	-4,73	0,011434	Succ
6	2,10	-28,11	29,99	1,79	-4,88	-4,63	0,004783	Succ
7	-29,99	-24,90	30,00	-4,35	-0,35	-3,82	3,245316	Fail
8	-27,50	22,62	-29,58	-4,64	-4,14	1,22	0,687886	Succ
9	-29,99	-20,38	30,00	-4,88	0,25	-3,91	3,245707	Fail
10	-23,76	29,98	-27,18	-4,92	-3,07	0,77	2,207561	Succ
				Best Fitness			0,004783	
				Average Fitness			2,227721	
				Success %			50%	

Table 2. Results of the FSA for Setup2

Run	Steer. Angle 1	Steer. Angle 2	Steer. Angle 3	Distance 1	Distance 2	Distance 3	Fitness	Result
1	9,22	-23,77	28,07	2,05	-4,98	-4,73	0,05836	Succ
2	9,89	-28	29,98	1,49	-4,58	-4,67	0,00233	Succ
3	-28,95	24,71	-21,23	-4,53	-4,43	1,35	0,510983	Succ
4	4,43	-27,46	30	1,74	-4,86	-4,61	0,008628	Succ
5	18,5	-21,38	28,38	1,78	-4,86	-4,63	0,009209	Succ
6	-13,62	-29,96	29,97	1,61	-4,96	-4,09	1,544968	Succ
7	-30	25,31	30	-4,6	-0,77	-3,17	3,262214	Fail
8	-29,99	-20,67	30	-4,05	-0,76	-3,74	3,278723	Fail
9	15,48	-23,29	27,54	1,83	-4,69	-4,84	0,010583	Succ
10	-29,51	29,99	-0,38	-4,73	-4,38	1,42	0,704797	Succ
				Best Fitness			0,00233	
				Average Fitness			0,939079	
				Success %			80%	

Figure 4 depicts a more or less classical 3-move solution for parallel parking that was rather expected from the optimization algorithm.

It is worth mentioning that the solution of figure 5 suggests that the specific parallel car park problem is in fact able to be solved with only two moves instead of three, a fact that is truly remarkable, and is attributed to the rather large space reserved for the parking place (8 meters) in the specific problem.

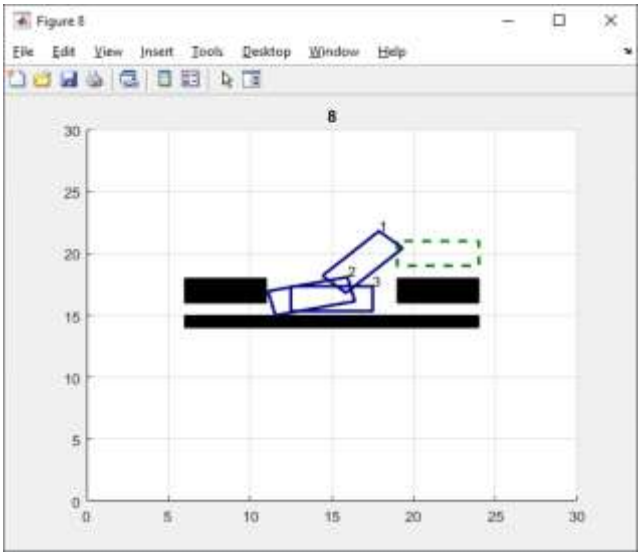

Fig.4 Successful classic parking solution found by FSA (3 moves)

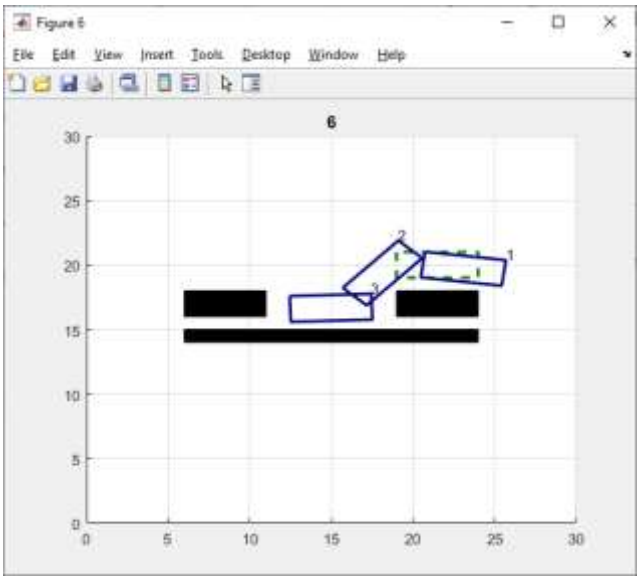

Fig.5 A rather two-move solution found by FSA

6. Conclusions

In this work an implementation of the Fish Swarm Algorithm (FSA) was used to address the problem of Parallel Car Parking. The results show that both the 3-moves Ackermann model adopted and the search ability of FSA are practically justified. FSA shows a very good 80% success rate that may get even better by increasing the algorithm's resources, in terms of the number of fish and the number of iterations. The solutions produced are comprehensible and to a large extent, expected. However, some of them are also very interesting, as they suggest that near optimal solutions may be achieved with less than the nominal 3 moves that comprise the encoded solutions. One major advantage of Evolutionary Algorithms, such as FSA, is that they are theoretically able to attack any kind of problem and therefore any kind of car parking problem, and not only the one described in this paper. So it is anticipated that FSA would exhibit similar

performance in other kinds of parking or other car movement problems, like for example, entering a ferry boat in reverse. Future work may include the application of different Evolutionary Computation methods on this problem, such as Genetic Algorithms or Particle Swarm Optimization. Also different parking problems can be addressed, with different topologies or in more narrow spaces, in order to test the generalization abilities of the proposed optimization method.

References:

[1] Marvy Badr Monir Mansour, Abdelrahman Said, Nour Eldeen Ahmed and Seif Sallam, "Autonomous Parallel Car Parking", in 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), London, UK, 2020, pp. 392-397, doi: 10.1109 / World S450073.2020.9210298.

[2] Laugier, I.E. "Motion Generation and Control for Parking an autonomous vehicle", France, IEEE, 1996.

[3] CNN (2003, September 1). Toyota unveils car that parks itself. Retrieved from https: //edition.cnn.com/2003/ TECH/ptech/09/01/toyota.prius.reut/

[4] Mays K. (2009, January 15) , "Up Close: Ford's Self-Parking System" retrieved from Cars.com: https://www.cars.com/articles/up-close-fords-self-parking-system-1420663258354/

[5] Xu J., Chen G., Xie M. "Vision-guided automated parking for smart car" in Intelligent Vehicles Symposium, 2000, IV, Proceedings of IEEE, pp. 725-730

[6] Wang. Y. and Zhu X. "A robust design of hybrid fuzzy controller with fuzzy decisions tree for autonomous intelligent parking system" in 2014 American Control Conference, pp. 5282-5287, IEEE press.

[7] Ji X., Wang J., Zhao Y., Lin Y., Zang L. & Li B. "Path planning and tracking for vehicle parallel parking based on preview BP neural network PID controller", Transactions of Tianjin University 21(3), pp. 199-208.

[8] Saleh M. Ismail M., & Riman N., "Enhanced algorithm for autonomous parallel parking of a car-like mobile robot", in 2016 IEEE International Multidisciplinary Conference on Engineering Technology (IMCET) pp. 191-195, IEEE press.

[9] Nakrani N. & Joshi M. "An Intelligent Fuzzy-based Hybrid Approach for Parallel Parking in Dynamic Environment" in Procedia Computer Science, 133, 2018, pp. 82-91.

[10] Rashid M. Rahman M., Islam M., Alwahedy O. & Abdulahi A. "Autonomous 4WD smart car parallel self-parking system using fuzzy logic controller" in American International Journal of Sciences and Engineering Research 2(2), 2019, pp. 1-31.

[11] X. L. Li, Z. J. Shao, J. X. Qian, "Optimizing method based on autonomous animats: Fish-swarm algorithm", System Engineering Theory and Practice 22 (11), (2002) pp.32-38 (in Chinese).

[12] Beni, G., Wang, J. (1993). Swarm Intelligence in Cellular Robotic Systems. In: Dario, P., Sandini, G., Aebischer, P. (eds) Robots and Biological Systems: Towards a New Bionics?. NATO ASI Series, vol 102. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-58069-7_38

[13] D. Yazdani, B. Saman, A. Sepas-Moghaddam, F. Mohammad-Kazemi, M. R. Meybodi, "A new algorithm based on improved artificial fish swarm algorithm for data clustering", International Journal of Artificial Intelligence 11 (13) (2013), pp. 1-29.

[14] X. Lei, H. Ouyang, L. Xu, "Image segmentation based on equivalent three-dimensional entropy method and artificial fish swarm optimization algorithm", Optical Engineering 57 (10) (2018) 103106.

[15] J. Zhu, C. Wang, Z. Hu, F. Kong, X. Liu, "Adaptive variational mode decomposition based on artificial fish swarm algorithm for fault diagnosis of rolling bearings", Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 231 (4) (2017), pp. 635-654.

[16] G. Zhou, Y. Li, Y.-C. He, X. Wang, M. Yu, "Artificial fish swarm based power allocation algorithm for mimo-ofdm relay underwater acoustic communication", IET Communications 12 (9) (2018), pp. 1079-1085.

[17] H. Shao, H. Jiang, H. Zhao, F.Wang, "A novel deep autoencoder feature learning method for rotating machinery fault diagnosis", Mechanical Systems and Signal Processing 95 (2017), pp. 187-204.

[18] W. Chen, Y.-Z. Feng, G.-F. Jia, H.-T. Zhao, "Application of artificial fish swarm algorithm for synchronous selection of wavelengths and spectral pretreatment methods in spectrometric analysis of beef adulteration", Food Analytical Methods 11 (8) (2018), pp. 2229-2236.

[19] K. B. Maji, R. Kar, D. Mandal, S. Ghoshal, "Optimal design of low power high gain and high speed cmos circuits using fish swarm optimization algorithm", International Journal of Machine Learning and Cybernetics 9 (5) (2018), pp. 771-786.

[20] H. Li, Y. Huang, S. Tian, "Risk probability predictions for coal enterprise infrastructure projects in countries along the belt and road initiative", International Journal of Industrial

Ergonomics 69 (2019), pp. 110-117.
[21] X. Zhou, Z. Wang, D. Li, H. Zhou, Y. Qin, J. Wang, "Guidance systematic error separation for mobile launch vehicles using artificial fish swarm algorithm", IEEE Access 7 (2019), pp. 31422-31434.
[22] C. Du, X. Sun, J. Zhou, Z. Dai, D. Yin, "Precision distribution method of navigation system based on improved artificial fish swarm algorithm", in: 2018 10th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Vol. 02, 2018, pp. 329-334.
[23] D. Mechta, S. Harous, "Prolonging wsn lifetime using a new scheme for sink moving based on artificial fish swarm algorithm", in: Proceedings of the Second International Conference on Advanced Wireless Information, Data, and Communication Technologies, ACM, 2017, pp. 1-5.
[24] M. Talha, M. S. Saeed, G. Mohiuddin, M. Ahmad, M. J. Nazar, N. Javaid, "Energy optimization in home energy management system using artificial fish swarm algorithm and genetic algorithm", 2018.
[25] Lynch, Kevin M., and Frank C. Park. Modern Robotics: Mechanics, Planning, and Control. Cambridge University Press, 2017
[26] Su, Bing, Liu. (2023). Analysis of the Car Model and Ackermann's Steering Geometry. Highlights in Science, Engineering and Technology, 37:1-13. doi: 10.54097/hset.v37i.6033.
[27] Deb, Kalyanmoy. Multi-objective optimization using evolutionary algorithms. Vol. 16. John Wiley & Sons, 2001.
[28] S. Kazarlis and V. Petridis, "Varying Fitness Functions in Genetic Algorithms: Studying the Rate of Increase of the Dynamic Penalty Terms," Proceedings of the 5th International Conference on Parallel Problem Solving from Nature (PPSN-V), Amsterdam, 27-30 September 1998, pp. 211-220.
[29] Mehdi Neshat, Ghodrat Sepidnam, Mehdi Sargolzaei, Adel Najaran Toosi , "Artificial fish swarm algorithm: a survey of the stateof-the-art, hybridization, combinatorial and indicative applications", Artif Intell Rev (2014) 42, pp. 965–997 DOI 10.1007/s10462-012-9342-2.
[30] Chun Fei, Ping Zhang, Jianping Li, "Motion Estimation based on Artificial Fish-Swarm in H.264/AVC Coding," WSEAS Transactions on Signal Processing, vol. 10, pp. 221-229, 2014.
[31] Naitik M. Nakrani, Maulin M. Joshi, "Fuzzy Based Adaptive Dimension Parking Algorithm Including Obstacle Avoidance for Autonomous Vehicle Parking," WSEAS Transactions on Computers, vol. 19, pp. 277-284, 2020, DOI:10.37394/23205.2020.19.33

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.