grey wolf optimizer

grey wolf optimizer is based on a social hierarchy where they are divided into alpha, beta, gamma, wolves encircle prey then attack them here prey refers to an optimal and the wolves keep changing their position in order to meet their prey (get an optimal solution). The rest wolves lead the others

Application

They can be used in self driving car where car refers to wolves and path refers to the prey so car determines which is the best path to take in order to reach the goal

Pseudocode

(1) First we need to initialise parameters and population

function GWO (wolves, num_iterations, bound)

wolf = initialise_population(objective_function, wolves)

alpha, beta delta = initialise()

(2) Next we run the code for num_iterations

for iteration in range num_iterations:

(3) We run the loop for each wolf

for wolf in wolves:

we evaluate fitness

fitness: objective_function (self.position)

update leader (alpha, beta, delta)

(4) Next we update the position of wolves and we update the fitness

update leader (alpha, beta, gamma)

if fitness < alpha.position

delta = beta

beta = alpha

alpha = wolf

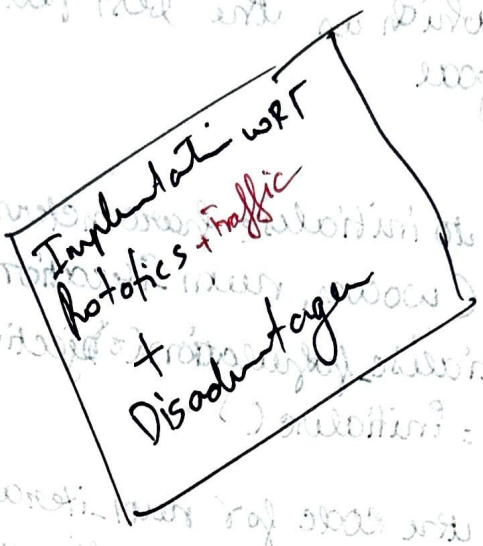
elif fitness < beta.position

delta = beta

beta = wolf

else:
delta = wolf

5) Next we need to update the position of wolf
updateposition() → to randomly generate values
{
a = a - (a * 2 iterations / max-iterations)
X = influence (alpha.position[i], position[i], a)
Y = influence (beta.position[i], position[i], a)
Z = influence (delta.position[i], position[i], a)
final = (X+Y+Z) / 3
return final
}



Implementation of Robotics
→ Sensor fusion and localization
- LIDAR: surrounding
  Cameras: visually
  IMU: track movement
- GWO helps us find best weights or setting for the sensors
- → randomly guesses the weights
  → weight → accerlation position
  → Keep adjusting the weights each iteration

Disadvantages

(1) GWO can sometimes settle on a solution too quickly thinking it's the best

(2) Sensitive to parameters
GWO has settings that control how it works if we don't set these parameters right it can slow down or not work well

(3) Lack of diverse
GWO uses the best wolves to guide the search but if they all end up in similar spots the search becomes less diverse

(4) Noisy Objective functions
In real world problems the data might be unclear or "noisy", GWO relies on accurate feedback to guide its search

(5) Not Ideal for all problems
GWO may not be the best choice for problems that involves things like decision or categories (like choosing the best schedule)

Output:

Iteration 1 Best Error = 1.12196589
Iteration 2 Best Error = 0.79265643
Iteration 3 Best Error = 0.79265646 33
Iteration 4 Best Error = 0.51283970 49
Iteration 5 Best Error = 0.512839704923
Iteration 6 Best Error = 0.5, 28397049 23 3.0838169

Optimal denor Weights : [0.7993339 7.18496409

O1P

Iteration 1 Error = 25.0

Iteration 2 Error = 20.0

Iteration 3 Error = 20.0

Iteration 4 Error = 20.0

Iteration 5 Error = 20.0

Optimal [+0.91830463 0.29821492]

22/11/24

10/10