OPTIMAL PATH FOR ROBOT MOVEMENT USING PSO

A Project Report

Submitted in the partial fulfillment of the requirements for

the award of the degree of

Bachelor of Technology
in
Department of Computer Science and Engineering

by

Manichand (2010030455)
Dileep Reddy (2010030416)
Harsha Vardhan (2010030173)
Prakash Raj (2010030533)

UNDER THE SUPERVISION OF

G.Madhukar Rao

Department of Computer Science and Engineering

K L University Hyderabad,

Aziz Nagar, Moinabad Road, Hyderabad – 500 075, Telangana, India.

March

DECLARATION

The mathematical programming -2 Report entitled “OPTIMAL PATH FOR ROBOT MOVEMENT USING PSO” is a record of Bonafede work of **Manichand (2010030455), Dileep Reddy(20100030416), Harsha Vardhan (2010030173), Prakash raj(2010030533**), submitted in partial fulfillment for the award of B.Tech in the Department of Computer Science and Engineering to the K L University, Hyderabad. The results embodied in this report have not been copied from any other Departments/ University/ Institute.

CERTIFICATE

This is to certify that the mathematical programming - 2 Report entitled “OPTIMAL PATH FOR ROBOT MOVEMENT USING PSO” is being Manichand(2010030455), Dileepreddy(20100030416) ,Harsha(2010030173),Prakashraj (2010030533), submitted in partial fulfillment for the award of B. Tech in CSE to the K L University, Hyderabad is a record of Bonafede work carried out under our guidance and supervision.

The results embodied in this report have not been copied from any other departments/ University/Institute.

Signature of the Supervisor

Signature of the HOD

Signature of the External Examiner

ACKNOWLEDGEMENT

First and foremost, we thank the lord almighty for all his grace & mercy showered upon us, for completing this project successfully.

We take grateful opportunity to thank our beloved Founder and Chairman who has given constant encouragement during our course and motivated us to do this project. We are grateful to our Principal **Dr. L. Koteswara Rao** who has been constantly bearing the torch for all the curricular activities undertaken by us.

We pay our grateful acknowledgement & sincere thanks to our Head of the Department **Dr. Chiranjeevi Manike** for his exemplary guidance, monitoring, and constant encouragement throughout the course of the project. We thank **G.Madhukar Rao** of our department who has supported throughout this project holding a position of supervisor.

We whole heartedly thank all the teaching and non-teaching staff of our department without whom we won't have made this project a reality. We would like to extend our sincere thanks especially to our parent, our family members and friends who have supported us to make this project a grand success.

ABSTRACT

robot motion planning with respect to two objectives, the shortest and smoothest path criteria. A Particle Swarm Optimization (PSO) algorithm is employed for global path planning, while the Probabilistic Roadmap method (PRM) is used for obstacle avoidance (local planning). The two objective functions are incorporated in the PSO equations in which the path smoothness is measured by the difference of the angles of the hypothetical lines connecting the robot's two successive positions to its goal. The PSO and PRM are combined by adding good PSO particles as auxiliary nodes to the random nodes generated by the PRM. The proposed algorithm is compared in path length and runtime with the mere PRM method searched by Dijkstra's algorithm, and the results showed that the generated paths are shorter and smoother and are calculated in less time.

TABLE OF CONENTS

INTRODUTION

In path planning, robot needs to navigate on a particular route whether the environment is familiar or not to the robot. The robot completed above type of tedious tasks efficiently and effectively without any human interruption. To cope up with such situation “Path Planning” term has been introduced. Now-a-days mobile robots are vastly used in many areas such as in military purposes, space research, emergency situations like fire hazard, medical use etc. During navigation of mobile robot various types of obstacles or hurdles comes across the robot and it needs to overcome those hurdles safely without collision and find the suitable path from source to goal point.

LITERATURE SURVEY

TITTLE	AUTHOR	Year of published	pros	cons
Particle Swarm Optimization in Global Path Planning for Swarm of Robots	Halder, Ritesh Kumar	2021	In this paper, a novel collision avoidance method was described. It uses PSO for optimizing the Ferguson splines.	if the situation in the robot space is changed, algorithm is restarted.
Optimal Trajectory Planning Analysis of Robot Manipulator Using PSO	gurjeet singh , Banga, V. K.	2021	This article developed to frame the inverse and forward kinematic model of 5 DOF and 6 DOF robot manipulator s velocity and acceleration increased through the proposed system.	the obstacle avoidance has been exhibited by PSO.
Path planning of mobile robot based on particle swarm optimization algorithm	Yang, Huimin, Lei Jiang, and Lijuan Wu	2021	The improved algorithm is tested in some test functions,and the simulation results show that the algorithm has high search speed and precision.	the algorithm converges prematurely, easily falls into local extremum,and reduces the convergence speed

Shortest Path Planning Algorithm – A Particle Swarm Optimization (PSO) Approach	Joshua T. Jegede, Hilary I. Okagbue and Pelumi E. Oguntunde	2020	The algorithm uses Particle Swarm Optimization approach to avoid convergence into a local minimum. With different experiments we show that the algorithm finds the shortest path in any known environment	No obstacles scattered in the environments to see how the algorithm gets the optimal path.
Hybrid metaheuristic approach for robot path planning in dynamic environment	Lina Bassem Amar, Wesam M. Jasim	2021	Some comparative results are presented on the basis of simulation results to illustrate the efficiency and the feasibility of the proposed algorithm. This approach supplies the optimal path with minimum number of iterations.	Not tested these three algorithms in a dynamic environment with moving goal.

HARDWARE REQUIRMENTS & SOFTWARE REQUIRMENTS

Hardware Requirements

We strongly recommend a computer fewer than 5 years old.

- Processor: Minimum 1 GHz; Recommended 2GHz or more
- Ethernet connection (LAN) OR a wireless adapter (Wi-Fi)
- Hard Drive: Minimum 512 GB; Recommended 1 TB or more
- Memory (RAM): Recommended 4 GB or above

Software Requirements

- Python
- Jupiter notepad

PROPOSED SYSTEM

A MPSO algorithm is used to find optimal path for the mobile robot . The proposed algorithm

(MPSO) is capable of effectively guiding a robot moving from start position to the goal position in complex environment and find optimum/shortest path without colliding any obstacles in the environment.

MPSO can rightfully be regarded as a good choice due to its convergence speed and robustness in global search

Let's us assume a few parameters first. You will find some new parameters, which I will describe later.
f: Objective function, Vi: Velocity of the particle or agent, A: Population of agents, W: Inertia weight,
C1: cognitive constant, U1, U2: random numbers, C2: social constant, Xi: Position of the particle or agent,
Pb: Personal Best, gb: global Best

The actual algorithm goes as below :

1. Create a 'population' of agents (particles) which is uniformly distributed over X.
2. Evaluate each particle's position considering the objective function(say the below function).

$$z=f(x, y)=\sin x2+\sin y2+\sin x\sin y$$

3. If a particle's present position is better than its previous best position, update it.
4. Find the best particle (according to the particle's last best places).
5. Update particles' velocities
6. Move particles to their new positions
7. Go to step 2 until the stopping criteria are satisfied.

IMPLEMENTATION

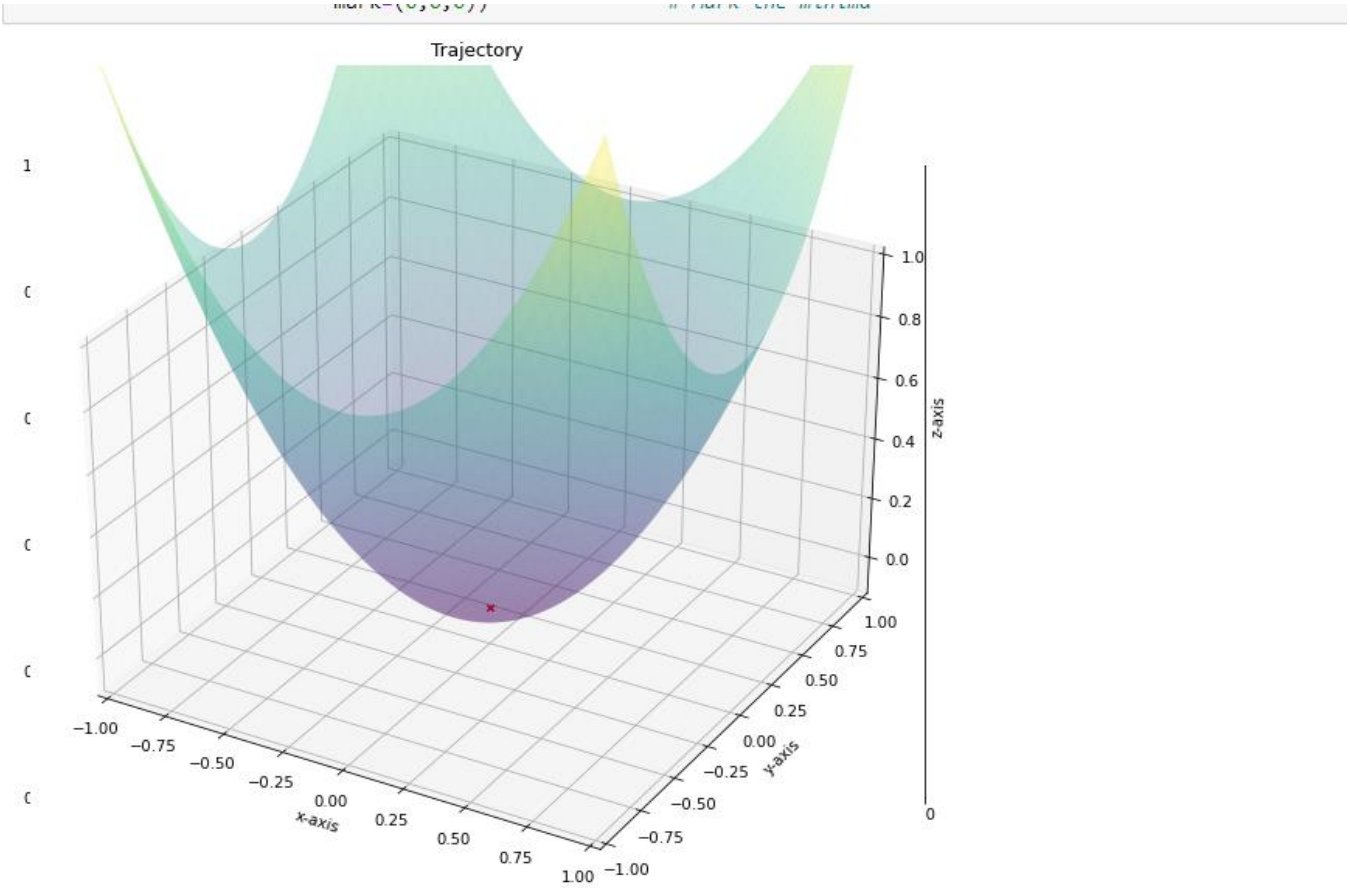
```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
from pyswarm import pso
from FringeSearch import FringeSearch
from ObjectiveFunction import ObjectiveFunction
import os
import math
# import ros stuff
import rospy
# import ros message
from geometry_msgs.msg import Point
from sensor_msgs.msg import LaserScan
from nav_msgs.msg import Odometry
from tf import transformations
from gazebo_msgs.msg import ModelState
from gazebo_msgs.srv import SetModelState
# import ros service
from std_srvs.srv import *

initial_position_ = Point()
initial_position_.x = rospy.get_param('start_x')
initial_position_.y = rospy.get_param('start_y')
initial_position_.z = 0
current_position_ = Point()
current_position_.x = rospy.get_param('current_x')
current_position_.y = rospy.get_param('current_y')
current_position_.z = 0
desired_position_ = Point()
desired_position_.x = rospy.get_param('goal_x')
desired_position_.y = rospy.get_param('goal_y')
desired_position_.z = 0
```

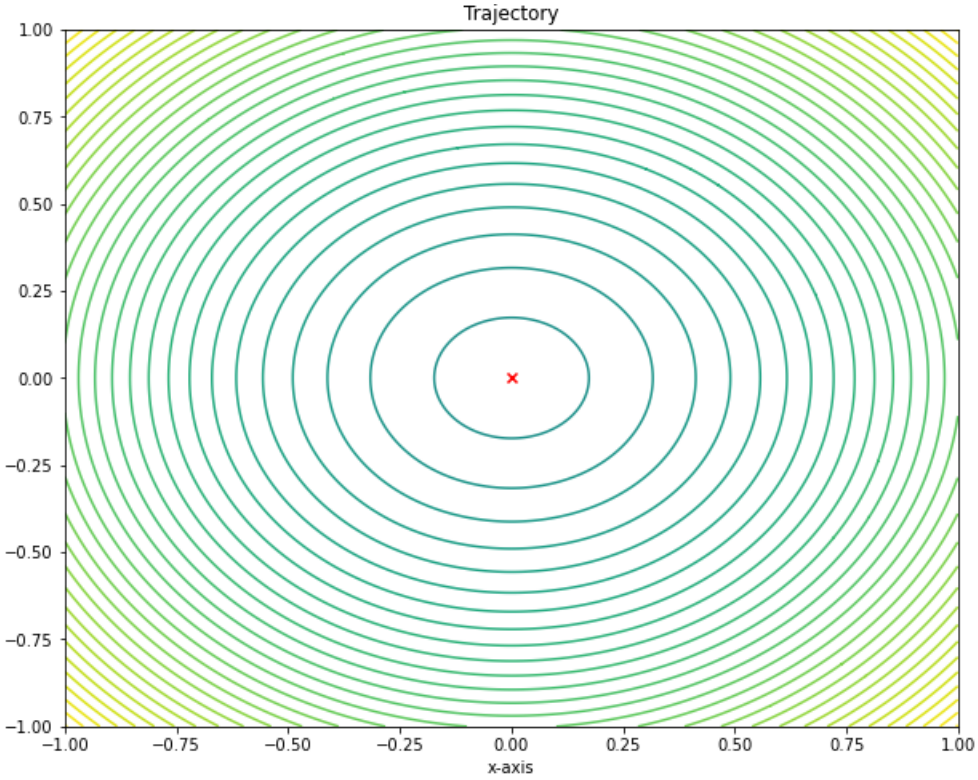
```
world = rospy.get_param('world')
path = os.path.dirname(__file__)
f = open(path+"/"+world+".txt", "r")
obstacles = f.readline()
f.close()

def main():
    rospy.init_node('psofs')

    sub_laser = rospy.Subscriber('/scan', LaserScan, clbk_laser)
    sub_odom = rospy.Subscriber('/odom', Odometry, clbk_odom)

    rospy.wait_for_service('/gazebo/set_model_state')
    srv_client_set_model_state = rospy.ServiceProxy('/gazebo/set_model_state',
SetModelState)

    # set robot position
    model_state = ModelState()
    model_state.model_name = 'tb3burger'
    model_state.pose.position.x = initial_position_.x
    model_state.pose.position.y = initial_position_.y
    resp = srv_client_set_model_state(model_state)

    rate_hz = 20
    rate = rospy.Rate(rate_hz)
    while not rospy.is_shutdown():
        PSOFS(initial_position_, obstacles, desired_position_)

        rate.sleep()

if __name__ == "__main__":
    main()
```

```
class PSOFS:
  # initialize robot_starting_point, world_obstacles, robot_destination
  def __init__(self, robot_initial_position, obstacles, destination):
    self.robot_initial_position = robot_initial_position
    self.obstacles = obstacles
    self.destination = destination
    self.algorithm()

  def algorithm(self):
    # create an empty list named path_points
    path_points = []

    # add robot_starting_point to path_points
    path_points.append(self.robot_initial_position)

    robot_current_position = self.robot_initial_position
    limit = 1

    while robot_current_position != self.destination:
      turning_point = []

      # pathfinding using PSO algorithm
      turning_point = pso(robot_curent_position)

      # if the distance between the robot's current position and any obstacle is less
than 1 meter
      if detect_obstacle() < limit:
        # set robot_current_position to the position where it almost collides with
obstacles
        robot_current_position = current_position_

        # pathfinding using Fringe Search algorithm
        turning_point = FringeSearch(robot_curent_position)
```

```
      # if the robot reaches the turning_point
      if current_position_ == turning_point:
        # add turning_point to path_point
        path_points.append(turning_point)

        robot_current_position = turning_point

    # add robot_destination to path_points
    path_points.append(self.destination)

    # create solution using path_points and world_obstacles
    solution = ObjectiveFunction().create_pareto_optimal_solution(self.obstacles,
path_points)

    # return solution, along with path_points
    solution_path = [solution, path_points]

    return solution_path
```

RESULT & DISCUSSION




Trajectory
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
x-axis

CONCLUSION AND FUTURE WORK

The most exciting part of PSO is there is a stable topology where particles are able to communicate with each other and increase the learning rate to achieve global optimum. The metaheuristic nature of this optimization algorithm gives us lots of opportunities as it optimizes a problem by iteratively trying to improve a candidate solution. Applicability of it will increase more with the ongoing research work in Ensemble Learning.

REFERENCES

[1] Halder, Ritesh Kumar. "Particle Swarm Optimization in Global Path Planning for Swarm of Robots." In Applying Particle Swarm Optimization, pp. 209-232. Springer, Cham, 2021.

[2] Banga, V. K. "Optimal Trajectory Planning Analysis of Robot Manipulator Using PSO." (2021).

[3] Yang, Huimin, Lei Jiang, and Lijuan Wu. "Path planning of mobile robot based on particle swarm optimization algorithm." In Journal of Physics: Conference Series, vol. 2093, no. 1, p. 012013. IOP Publishing, 2021.

[4] Yang, Huimin, Lei Jiang, and Lijuan Wu. "Path planning of mobile robot based on particle swarm optimization algorithm." In *Journal of Physics: Conference Series*, vol. 2093, no. 1, p. 012013. IOP Publishing, 2021.

[5] Amar, Lina Basem, and Wesam M. Jasim. "Hybrid metaheuristic approach for robot path planning in dynamic environment." Bulletin of Electrical Engineering and Informatics 10, no. 4 (2021): 2152-2162.