International Journal of Soft Computing and Engineering (IJSCE)
ISSN: 2231-2307, Volume-2 Issue-2, May 2012

Artificial Intelligence in Robot Path Planning

Yogita Gigras, Kusum Gupta

Abstract*—Mobile robot path planning problem is an important combinational content of artificial intelligence and robotics. Its mission is to be independently movement from the starting point to the target point make robots in their work environment while satisfying certain constraints. Constraint conditions are as follows: not a collision with known and unknown obstacles, as far as possible away from the obstacle, sports the shortest path, the shortest time, robot-consuming energy minimization and so on. In essence, the mobile robot path planning problem can be seen as a conditional constraint optimization problem. To overcome this problem, ant colony optimization algorithm is used.

Index Terms*— Particle Swarm Optimization (PSO), Genetic Algorithm(GA), Tabu Search, Simulated Annealing (SA), Reactive Search Optimization (RSO), proportional–integral–derivative(PID).

I. INTRODUCTION

Robot path planning is about finding a collision free motion from one position to another. Efficient algorithms for solving problems of this type have important applications in areas such as: industrial robotics, computer animation, drug design, and automated surveillance [1,2]. By representing synthetic, simulated humans as robots, we can use motion planning algorithms to develop convincing computer generated animation. There are many traditional techniques used in past in robot control such as PID. Problem with PID control is that they perform process efficiently over very limited range of environment. It is very difficult to have highly accurate performance especially at high speed of processes. This is because of PID control i.e. PID is linear and not suitable for non-linear system with varying dynamic parameters and PID requires precise knowledge of dynamic model. This may explain the dominant role of soft computing techniques in robotics. During the last four decades, researchers have proposed many techniques for control and automation. There are various step involved in designing of control system. These are modelling, analysis, simulation, implementation and verification. In conventional/traditional techniques of control, the prime objectives had been precision and uncertainty. However, in soft computing, the precision and certainty can be achieved by techniques of fuzzy logic, neural network, evolutionary algorithm, and hybrid. The main emphasis of the paper is to explore the efficient and accurate procedure based on soft-computing algorithm to provide the online learning mechanism which performs better in dynamic, unstructured environment of robot [3]. Many techniques are used for this:-

Manuscript received on April 26, 2012.

Yogita Gigras, Computer Science and information technology, ITM University, Gurgaon, India,
09718255199, (Email:gigras.yogita@gmail.com).
Kusum Gupta, Computer Science, Banasthali Vidyapith, Banasthali, India , 09829191957, (E-mail:gupta_kusum@yahoo.com).

A. *Particle Swarm Optimization (PSO)*

Particle Swarm Optimization (PSO) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. PSO optimizes a problem by having a population of candidate solutions, with dubbed particles, and moving these particles around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position and is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions. PSO is a metaheuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, metaheuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time.

B. *Genetic algorithm (GA)*

A genetic algorithm (GA) is a search heuristic that mimics the process of natural evolution. This heuristic is routinely used to generate useful solutions to optimization and search problems. Genetic algorithms belong to the larger class of evolutionary algorithms (EA), which generate solutions to optimization problems using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. In a genetic algorithm, a population of strings called chromosomes or the genotype of the genome, which encodes candidate solutions called individuals, creatures, or phenotypes to an optimization problem, evolves toward better solutions. Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and happens in generations. In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), and modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of generations, a satisfactory solution may or may not have been reached.

C. Tabu Search

Tabu Search is a local search method used for mathematical optimization. Local searches take a potential solution to a problem and check its immediate neighbors in the hope of finding an improved solution. Local search methods have a tendency to become stuck in suboptimal regions or on plateaus where many solutions are equally fit. Tabu search enhances the performance of these techniques by using memory structures that describe the visited solutions or user-provided sets of rule. If a potential solution has been previously visited within a certain short-term period or if it has violated a rule, it is marked as "taboo" so that the algorithm does not consider that possibility repeatedly.

D. Simulated Annealing (SA)

Simulated annealing (SA) is a generic probabilistic metaheuristic for the global optimization problem of locating a good approximation to the global optimum of a given function in a large search space. It is often used when the search space is discrete. For certain problems, simulated annealing may be more efficient than exhaustive enumeration — provided that the goal is merely to find an acceptably good solution in a fixed amount of time, rather than the best possible solution. The name and inspiration come from annealing in metallurgy, a technique involving heating and controlled cooling of a material to increase the size of its crystals and reduce their defects. The heat causes the atoms to become unstuck from their initial positions (a local minimum of the internal energy) and wander randomly through states of higher energy; the slow cooling gives them more chances of finding configurations with lower internal energy than the initial one.

By analogy with this physical process, each step of the SA algorithm attempts to replace the current solution by a random solution (chosen according to a *candidate distribution*, often constructed to sample from solutions near the current solution). The new solution may then be accepted with a probability that depends both on the difference between the corresponding function values and also on a global parameter *T* (called the *temperature*), that is gradually decreased during the process. The dependency is such that the choice between the previous and current solution is almost random when *T* is large, but increasingly selects the better or "downhill" solution (for a minimization problem) as *T* goes to zero. The allowance for "uphill" moves potentially saves the method from becoming stuck at local optima—which are the bane of greedier methods.

E. Reactive Search Optimization (RSO)

Reactive Search Optimization (RSO) defines local-search heuristics based on machine learning, a family of optimization algorithms based on the local search techniques. It refers to a class of heuristics that automatically adjust their working parameters during the optimization phase. Reactive Search Optimization (RSO), like all local search techniques, is applied to the problem of finding the optimal configuration of a system; such configuration is usually composed of continuously or discretely varying parameters, while the optimality criterion is a numerical value associated to each configuration. In most cases, an optimization problem can be reduced to finding the (global) minimum of a function whose arguments are the configuration parameters, seen as free variables in the function's domain space.

Reactive Search Optimization advocates the integration of sub-symbolic machine learning techniques into search heuristics for solving complex optimization problems. The word *reactive* hints at a ready response to events during the search through an internal *feedback loop for online self-tuning and dynamic* adaptation. *In Reactive Search the past history of the search and the knowledge accumulated while* moving in the configuration space is used for self-adaptation in an autonomic manner: the algorithm maintains the internal flexibility needed to address different situations during the search, but the adaptation is automated, and executed while the algorithm runs on a single instance and reflects on its past experience.

F. Ant colony algorithms

In the natural world, ants (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep traveling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food.

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained.

Various approaches to overcome the problem of ACO i.e mitigating stagnation which include:- evaporation, agingand pheromone smoothing .

The Approaches to alleviate stagnation is pheromone control. Pheromone control adopts several approaches to reduce the influence from past experience and encourage the exploration of new paths that are non-optimal.

(1) Evaporation:- To reduce the effect of past experience, an approach called evaporation is used in conjunction in optimal path from being excessively high and preventing ants from exploring the other paths. In each iteration, the pheromone value τ_{ij} in all edges are decremented by a factor *p* such that
$\tau_{ij} \leftarrow \tau_{ij}$ (1-*p*)

(2) Aging:- A past experience can also be reduced by controlling the amount of pheromone deposited for each ant according to its age. This approach is known as aging. In aging, an ant deposits lesser and lesser amount of pheromone as it moves from one obstacle to other obstacle. Aging is based on the rationale that "old" ants are less successful in locating the optimal paths since they take longer time to reach their destination. Both aging and evaporation encourage discoveries of new paths that are previously non-optimal.

(3) Limiting and smoothing pheromone:- Limiting the amount of pheromone in every path, by placing an upper bound on the amount of pheromone for every edge(i,j), the preference of an ant for optimal paths over non-optimal path is reduced. This approach prevents the situation of generating a dominant path. A variation of such an approach is called pheromone smoothing.

II. PROPOSED SOLUTION FOR ROBOTIC PATH PLANNING

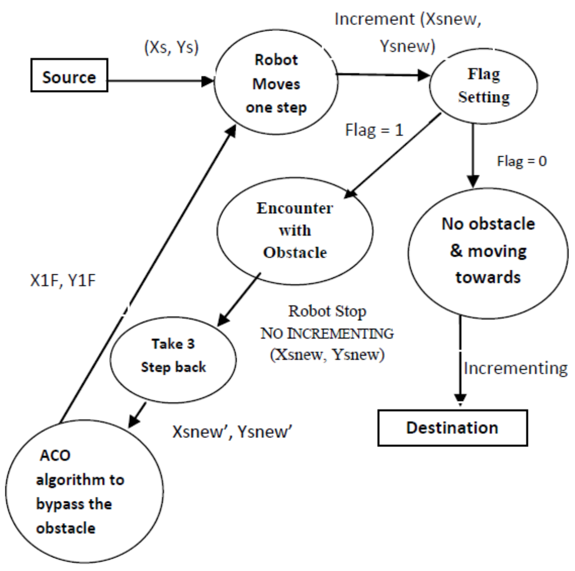

Fig1: Layout of robot path planning

A. Source

Robot start walking from source point(X_s,Y_s) and it is fixed.

B. Robot Moves one step

From source point(X_s,Y_s), Robot is moving towards destination point by taking one step a head and change the value of (X_s,Y_s) to (X_{snew},Y_{snew}) by using the below equation:-

$$X_{snew=} X_{prev}+step*\cos(\theta) \quad (1)$$

$$Y_{snew=} Y_{prev}+step*\sin(\theta) \quad (2)$$

Where as X_{prev},Y_{prev} denotes where the robot is currently situated in which we add the step size multiplied by cos(ɵ)and sin(ɵ) respectively which will gives robot's next position. Where ɵ is dynamic angle and it can be calculated by:-

$$\Theta = \mathrm{Tan}^{-1}(Xprev/\ Y_{prev}) \quad (3)$$

C. Flag Setting

Robot see the value of the fag, if its value is zero means it indicate that there is no obstacle and robot can take a one step ahead by using the process 2.2 and can reach to the destination point.

D. Encounter with obstacle

Whenever the robot encounter with obstacle, it has to stop moving means there is no increase in the step size and robot has to take three steps back. In our proposed work, twenty obstacle are generated randomly which is of rectangular shape. Number of obstacles is fixed which is a constraint in our work.

Obstacle (oopsV) can be generated by the following pseudo code:-

```
oopsV=20;
x=100*rand(1,oopsV);
y=100*rand(1,oopsV);
l=10*rand(1,oopsV);
w=10*rand(1,oopsV);
for m=1:oopsV
plot([x(1,m) x(1,m)+w(1,m)], [y(1,m) y(1,m)]);
plot([x(1,m) x(1,m)], [y(1,m) y(1,m)+l(1,m)]);
plot[x(1,m)  x(1,m)+w(1,m)],[y(1,m)+l(1,m)
y(1,m)+l(1,m)]);
plot([x(1,m)+w(1,m)x(1,m)+w(1,m)],[y(1,m)
y(1,m)+l(1,m)]);
end
```

In this pseudo code initialize the number of obstacles oopsV=20 and obstacles are generated in the moving space of 100*100 and whose length and width varies between 0 1o 10 dynamically.

E. Take three step back

Whenever the robot encounter with obstacle, robot stop moving and take three step back by using the following equation:-

$$X_{snew=} X_{prev}-3*step*\cos(\theta) \quad (4)$$

$$Y_{snew=} Y_{prev}-3*step*\sin(\theta) \quad (5)$$

$$\Theta = \mathrm{Tan}^{-1}(Xprev/\ Y_{prev}) \quad (6)$$

F. Destination

Finally robot has to reach at the point (X_T,Y_T), which is fixed. Robot has to bypass the obstacle and by following optimal path has to reach to target point.

G. Apply the ACO algorithm to bypass the obstacle

ACO is a metaheuristic algorithm inspired by the real ant for the forage for food. ACO is applied to the problems which can be described by the graphs so that feasible solution can be expressed in terms of paths on the graph. It was first applied to TSP. Among the feasible paths, ACO is used to find out the optimal one i.e. locally or globally optimal. This algorithm is implemented in two steps. In first step, the edge is selected on the basis of probability formula. Assume that ant k is located at node i, uses the pheromone τ_{ij} deposited on the edge (i,j) to compute the probability of choosing next node

$$\text{Pij} = \begin{cases} \dfrac{\tau_{ij^{\alpha}}}{\sum_{j \varepsilon N_{i^{(k)}}} \tau_{ij^{\alpha}}} \; if \;\; j \,\varepsilon\, N_{i^{(k)}} \\ 0 \qquad\qquad otherwise \end{cases} \tag{7}$$

Where α denotes the degree of importance of pheromone trail and $N_i^{(k)}$ indicates the set of neighbour of ant k when located at node i expect the predecessor node i.e. the last node visited by ant k. This will prevent the ant k for returning to the same node. An ant travels from node to node until it reaches to the destination node and come back to the source node.

In second step, once all the ants complete their tour, then global optimization of the pheromone trail takes place.

$$\tau_{ij} = (1-\rho)\,\rho + \sum_{k=1}^{N} \Delta\tau_{ij}^{(k)} \tag{8}$$

Where $\rho\ \varepsilon\ (0,1]$ is the evaporation rate and $\Delta\tau_{ij^{(k)}}$ is the amount of pheromone deposited on the edge (i,j) selected by the best ant k. The aim of pheromone updating is to increase the pheromone value associated with optimal path. The pheromone deposited on arc (i, j) by the best ant k is $\Delta\tau_{ij^{(k)}}$.

Where

$$\Delta\tau_{ij^{(k)}} = \frac{Q}{L_k} \tag{9}$$

Here Q is a constant and L_k is the length of the path traversed by the best ant k. This equation is also implemented as:-

$$\Delta\tau_{ij}^{(k)} = \begin{cases} \dfrac{\varsigma f_{best}}{f_{worst}} \; if\ (i,j)\,\varepsilon\ global\ best\ tour \\ 0 \qquad otherwise \end{cases} \tag{10}$$

III. CONCLUSION

Ant colony optimization is to be applied for robot –motion control such as navigation and obstacle avoidance in an efficient manner. From this, money can be saved and reliability can be increased by allowing them to adapt themselves according to the environment without further programming. Ant colony optimization (ACO) takes inspiration from the foraging behavior of ant species. These ants deposit pheromone on the ground in order to mark some favorable path.

REFERENCES

1. Yao-hong Qu, Quan Pan, Jian-guo Yan, “Flight Path Planning of UAV Based on Heuristically Search and Genetic Algorithms”, Annual Conference of IEEE on Industrial Electronics Society, (IECON),pp:5,2005.
2. Chih-Lyang Hwang, *Member*, *IEEE*, and Li-Jui Chang, “Internet-Based Smart-Space Navigation of a Car-Like Wheeled Robot Using Fuzzy-Neural Adaptive Control”, IEEE Transactions on Fuzzy Systems, pp: 1271 – 1284,2008
3. Abdullah Zawawi MOHAMED, Sang Heon LEE, Mahfuz AZIZ, Hung Yao HSU, Wahid Md FERDOUS, “A Proposal on Development of Intelligent PSO Based Path Planning and Image Based Obstacle Avoidance for Real Multi Agents Robotics System Application”, International Conference on Electronic Computer Technology (ICECT), pp: 128 – 132, 2010.

AUTHORS PROFILE

Yogita Gigras ,Assistant Professor in ITM University.She had done engineering from Babu Banarasi Das National Institute of Technology and Management affiliated to Uttar .Pradesh Technical University Lucknow and masters from Banasthali Vidyaphith, Rajasthan.
.

Dr. Kusum Gupta, Associate Professor in Banasthali Vidyapith University. She has 17 years of teaching experience in banasthali Vidyapith, Rajasthan. Her research areas are soft computing and algorithm
.