Autonomous Ship Manoeuvring Planning based on the Ant Colony Optimization Algorithm

J.M. Benitez, Juan F. Jimenez, Jose M. Giron-Sierra

Dep. ACYA, Fac. Fisicas, Universidad Complutense de Madrid
Av. Complutense s/n, 28040 Madrid, Spain
gironsi@dacya.ucm.es

1.1

Abstract: This paper presents some results achieved from a preliminary study on the use of the Ant Colony Algorithm Extension, including Cellular Automata, to plan feasible optimal or suboptimal trajectories for an autonomous ship manoeuvring from a starting point with a certain attitude to another not far location with a desired attitude..

1. INTRODUCTION

This paper presents some results achieved from a preliminary study on the use of the Ant Colony Algorithm to plan feasible optimal or suboptimal trajectories for an autonomous ship manoeuvring. The scenario, for this preliminary work, comprises only open sea manoeuvres. The goal involves obtaining the least time consuming ship trajectory between to points, departing from the start point with arbitrary initial speed and attitude values and arriving to the end point with predefined speed and attitude values.

The specific dynamic of the ship imposes typical restrictions to its manoeuvrability. In the present case, the non-holonomicity, the rate speed/turn radius, and the imposed forward-only propulsion of the ship make up the main restrictions to the ship movement. For long distances, the problem could be tackled as a classical navigation problem, in which, for the most part of the ship trajectory, techniques such inertial navigation should be enough. The problem arises at short distances when it becomes a manoeuvring problem. In this case to obtain an optimal, --in some cases just a feasible--, trajectory could be a difficult problem.

In recent years, several innovative optimisation techniques, based on heuristic search methods have been developed and proved in very different scenarios. Among them, the so called bioinspired algorithms, such the Ant Colony Optimisation or the Artificial Bee Colony Algorithm result particularly attractive by their capacity to solve complex optimisation problems in which, other classical techniques are unfeasible or difficult to implement.

The aim of the present work is to prove the viability of one of these techniques to obtain the trajectory of an autonomous ship in the manoeuvring scenario described above. To accomplish this goal, a simplified dynamical model of a ship, considering only three degrees of freedom (surge, sway and yaw) was employed. The propulsion was modelled as a trimmable waterjet system, which plays also the role of the rudder. Both, speed and course are controlled by a classical PID system, which stabilizes the course and speed, according to preset values of the PID constants.

The standard 6-DOF model employed to describe ship movement, see for example Lloyd (1998), has been simplified to a 3-DOF model. The selected equations are thefollowing:

$$(m + a_{11})\ddot{x}_1 + b_{11}\dot{x}_1 = F1 \quad (1)$$

$$(m + a_{22})\ddot{x}_2 + b_{22}\dot{x}_2 = F2 \quad (2)$$

$$(I_{66} + a_{66})\ddot{x}_6 + b_{66}\dot{x}_6 = F6 \quad (3)$$

where suffix 1 represents surge, suffix 2 sway, and suffix 6 yaw. The damping coeficients and added masses of these equations were chosen proportional to the speed. The remaining constant coefficients were chosen just to fullfil a reasonable ship behaviour. The ship actuators are waterjets for both, propulsion and steering,with limits imposed to the waterjet orientation and thrust. A PD feedback control system has been implemented to achieve speed and attitude set points.

Ant Colony Optimisation algorithms are based in the way in which ants are capable of finding the shortest path from a food source to their nest. Ants deposit a certain amount of pheromone while walking. When any ant searches a path to follow in its search for food, it prefers, in a probabilistic sense, the trails rich on pheromona. As far as shorter paths can be followed faster, the shorter the path the larger the number of ants that cover it by unit time. As a result, the shortest (optimum) path becomes more and richer on pheromona, and more a more ants follow it. Although the process tends to converge with time to the best way found by

the ants, the probabilistic nature of the ants path election, makes easier to avoid get trapped in local minima far from the optimal solution and it allows, once the algorithm has found a feasible solution, to improved it towards the optimum.

In the robotics context it is usual to divide the problem of robot motions into two steps: first to get an optimal path (path planning), according with certain cost criteria, and then making the robot follow the path (path following). It may be difficult for the robot to follow the path, due to the dynamic characteristics and restrictions of the robot itself.

In our case we propose to solve the motion problem in one step, so the path planning includes the robot dynamics and restrictions.

Now, the original ant colony algorithm doesn't consider any particular dynamics for the ants' movements. It just assigns an elapsed time proportional to the length covered by the ant in its movement towards the goal. Our contribution here is to add dynamics -the dynamics of the ship- and restrictions to the algorithm. In this way the trajectories obtained are feasible. The ants behave as ships.

In the original algorithm, the ant colony is composed by an arbitrary number of ants. They are sent from the nest, which represent the starting point of the quest, in random directions. The ants follow straight paths until they reach an obstacle, then a new random direction is selected. When any ant reaches the goal, it marks the track followed from the nest to the goal.

2. CELLULAR AUTOMATA TO START

Due to the nature of the manoeuvring problem, the space of search is really large to find not only the optimal but also a feasible solution, following a whole random search. Therefore we propose a way to guide the search. The idea is to use cellular automata.

The cellular automata perform a rough discretisation of the continuous search space of the problem, and provides a collection of recommended ship attitudes without any explicit reference to ship dynamics. These attitudes will be the starting point for the Ant Colony Algorithm to calculate continuous trajectories, taking into account the restrictions imposed by the ship dynamics. CA are a set of automata disposed on a squared matrix. In our case, each matrix's element corresponds to a 10x10m portion (a cell) of a square region. This square region is assigned to a particular ship maneouvre. The center of that matrix correspond to the cell containing the goal. The ship initial position is always, for convenience, located at the origin (0,0), with attitude 90o. Equations 4 and 5 describe how to calculate the CA dimension and the element (row, col) representing the goal region.

$$L = 2 * \text{ceil}(\max(|x|, |y|) / 10) + 1 \quad (4)$$

$$\text{row} = \text{col} = \text{fix}((L + 1)/ 2) \quad (5)$$

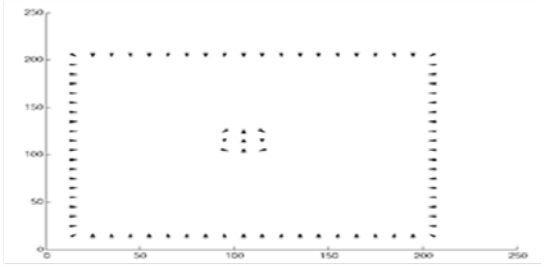
Each automaton represents an attitude value for its cell. Every automaton is an unitary vector. Their orientations in degrees from 0o to 359o with 1o precision are the possible states of the automaton. There are two kinds of automata, fixed (FA) and variable (VA).

In our case the state of an automaton is the sum of the orientations of their nearest neighbors

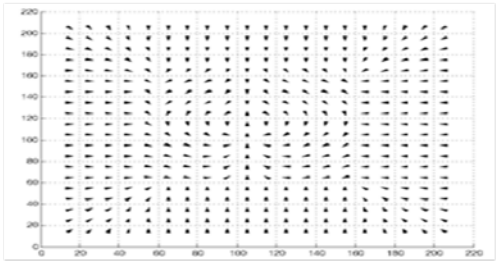
.

The goal is defined as fixed automaton with the goal attitude. Also the adjacent's automata are fixed too, with attitudes that draw a vector field similar to a magnetic field, with closed field lines. The matrix's bounds are initialized with attitude pointing to the goal cell. An example of automata initial states is shown in Fig. 1; where the cells containing fixed automata are marked with an arrow in the direction of the automaton state and the remaining automata are initialised with a (0,0) state and are represented as empty cells. CA evolve until they reach a stable configuration or an iteration limit.

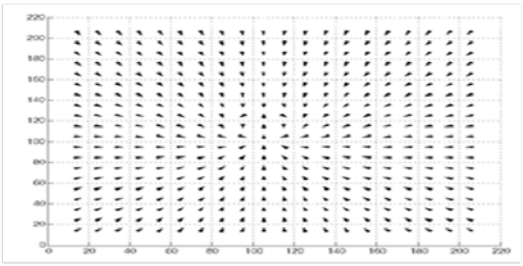
Let us put a sequence of figures, following the evolution of the computations:



start



In the middle



final

1.2 Fig. 1. Steps of the Cellular Automaton Evolution

3. EXTENSION OF THE ANT COLONY OPTIMIZATION

In the original algorithm, the ant colony is composed by an arbitrary number of ants. They depart from the nest in random directions. The ants follow straight paths until they reach an obstacle, then a new random direction is selected. When any ant reaches the goal, it marks the track followed from the nest to the goal.

In our case, the original algorithm has been extended to deal with the optimization problem at hand. Let us describe the three extensions we introduced.

3.1. First Extension.

Classical ACO uses graphs to model the problem, because ants will need a graph to perform the search. In our case, graphs are substituted by feasible trajectories generated from the dynamic equations of the ship. This extension could be applied to other dymamical systems, just changing the equations. In the search, some exploration trajectories do not find any end. As soon a trajectory hitting the target is found, the algorithm discard endless trajectories. This is an intrinsic feature of the parallel searching process of common to all swarm algorithms.

3.2. Second Extension.

The classical ACO can be applied only to finite search spaces, because new solutions are generated by combining small pieces of older solutions, this is called emergent behavior. In a previous work we point how to deal with infinite search, based on the behaviour of real ants The recipe is don't try to apply the ACO to the whole search space, but only to a finite region of it. This means that two separate processes are needed. The first one is the classical ACO applied to the solutions already discovered. The second one provides new solutions to the ACO by new explorations of the search space. In this way we employ two kinds of ants: The classical ACO ants (foragers ants) and the ants in charge of the exploration process (explorer ants). Each time our algorithm finds a better solution, it restricts the exploration space. Since solutions improve along the process, the exploration space shrinks gradually. With this strategy the method is able to deal with an infinite state search.

3.3. Third Extension.

Quality scores are assigned to solutions. The score is based on a objective function to be optimised. This approach forces to use some kind of criterion to discard solutions, in ACO this is called pheromone evaporation factor, because if the solution's pool became too large the combination process could fail. There is another reason to use that pheromone's evaporation: to avoid local minima. Instead of pheromone evaporation, we try to avoid local minima allowing the explorer ants not to follow previous solutions -pheromone- , so they have the chance to discover new solutions and escape from local minima.

Let us take a snapshot of the process. We have several solutions and we can compute a mean of objective function values. With this mean we can discard the worse solutions. In this way we are sure that every new solutions added to solutions' pool push outside the worse solution's. Also the mean improves along the process. In other words, score employed is dynamic instead of static. So there is no need of pheromone factor or criterion to discard worst solutions - local minima- the whole system will discard them by an evolutionary process.

The Ant Colony Extended (ACE) starts by sending explorer ants at regular time intervals. Once any of those ants find a solution ACE will change its state and starts the swarm searching process. This process mixes explorer and foragers ants, as new solutions are discovered the search space will be restrained. Figures 2 to 4 illustrate an example of the process.

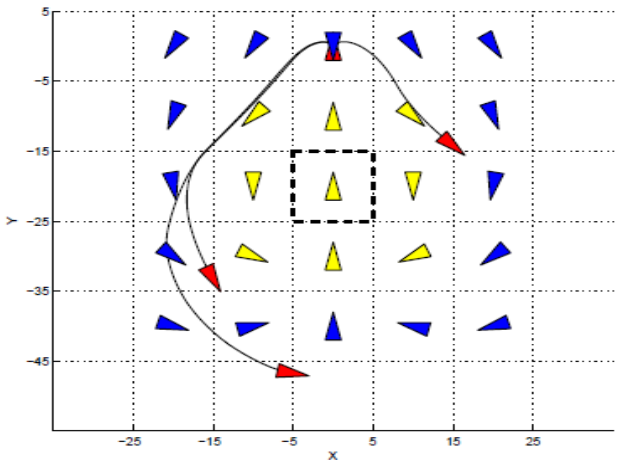


Fig. 2. Ant Colony Extended initial state. Goal automaton: orientation 90º, position (0m,-20m) final speed unrestricted

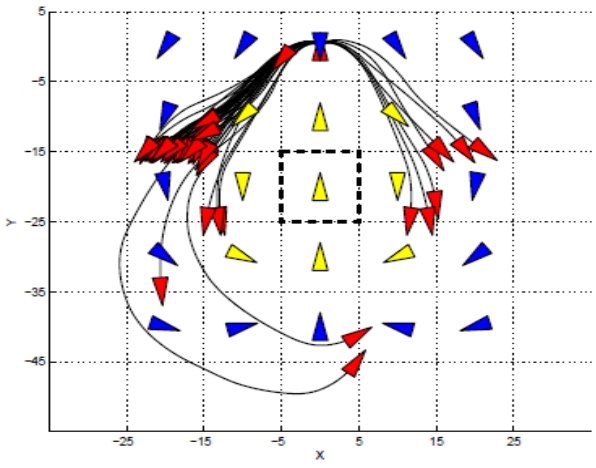


Fig. 3. Ant Colony Extended early stage of the search process. 90º, position (0m,-20m) final speed unrestricted

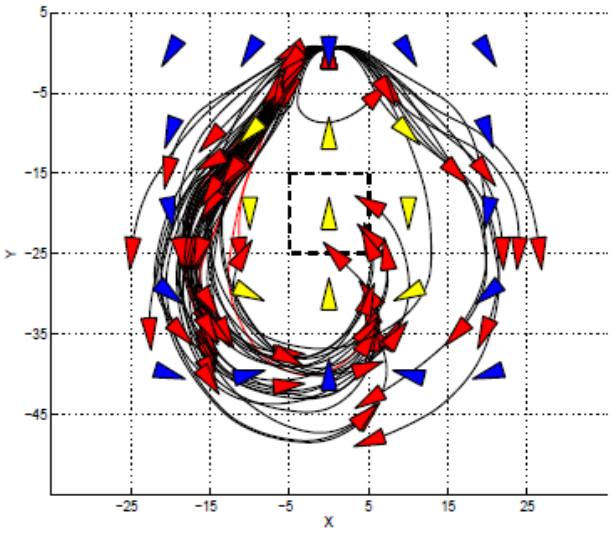


Fig. 4. Ant Colony Extended advanced stage of the search process. 90º, position (0m,-20m) final speed unrestricted

4. RESULTS

4.1. A Simple Experiment.

Next figures show the progress of the evolution for a simple experiment.

Figure 5 shows an initial phase of the search. The ants are going to pass two cells (the cell occupy 4 squares in the Matlab grid).

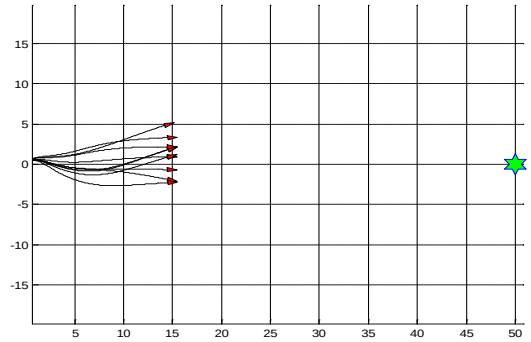


Fig. 5. First ants in a simple experiment

Next figure shows two consecutive generations of ants.

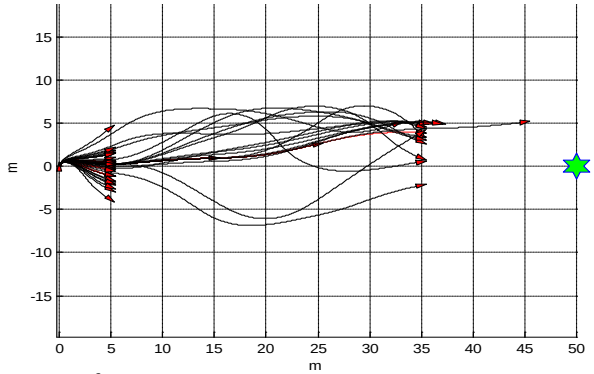


Fig. 6. After 2 generations

Next figure shows another more advanced stage of the search.

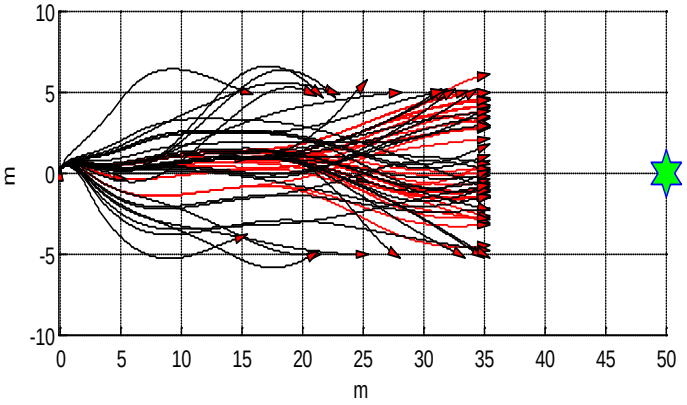


Fig. 7. After several generations

4.2. Single More Difficult Experiment

In a single experiment we alllow the algorithm to run for 1000 iterations with a limit of 200 ants at the same time.

When an ant arrives to the target the errors permited are a deviation of 1 knot in speed and 4o in attitude. The initial ship speed is zero. Figure 8 illustrates a single experiment with the next goal:

position = (0,− 20), attitude = 90º, speed = unrestricted.

The first solutions are the outer trajectiories in the figure. As the process continues, better solutions are found: the inner trayectories.

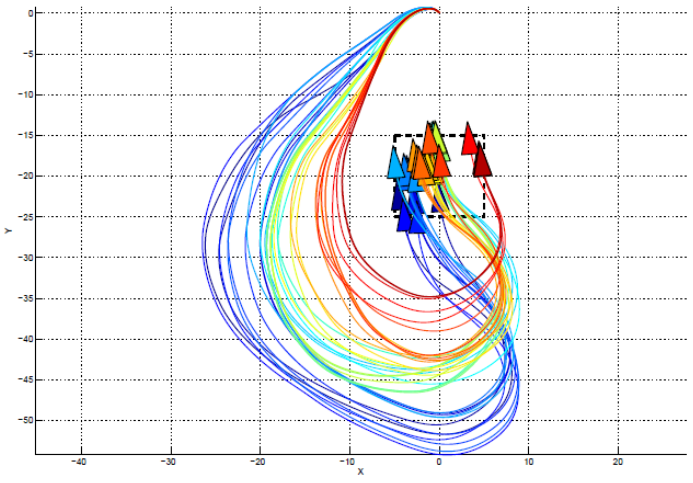


Fig. 8. Trajectories obtained for a single experiment

4.3. A Batch of Experiments

To test the convergence of the method towards the optimal solution, a batch of 100 experiments has been done. All experiments have the same goal and the same initial conditions. It has been observed that every experiment gets a different near-to-optimal solution.

Figure 7 shows the best 20 trajectories obtained, for the same goal as the single experiment described above. Obviously, in this particular scenario there are two symmetric optimal solutions –one turning left and the other turning right–. The ACE is able to approach both of them. Figure 9 shows the statistics for the batch of experiments; each column corresponds to a single experiment. The upper graphic shows how many solutions each single experiment has discovered. The middle one shows the time invested by the best solution; notice that all of them are between 9 and 10 seconds . And the lower graphic shows the time invested by the worst solution; notice that worst solutions -first discovered solutions- could be very far from optimal but the algorithm can still converge towards the optimum.

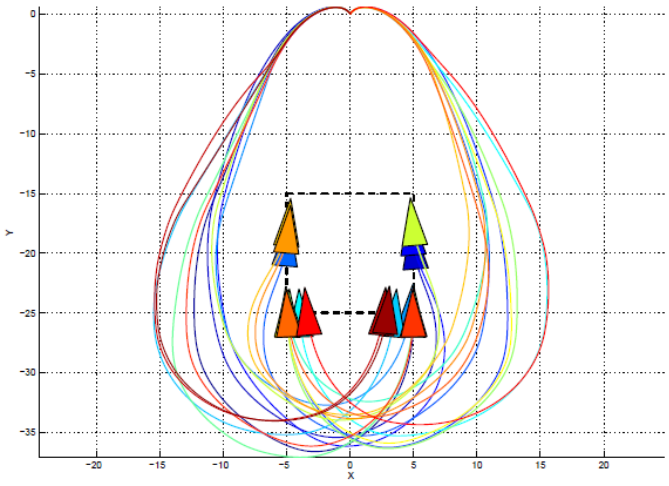


Fig. 9. The twenty best trajectories obtained during the batch of experiments

5. CONCLUSION

In this paper the short distance manoeuvring of a ship, having specified attitudes at the start and at the goal, has been studied. The paper proposes an extension of the Ant Colony Optimization algorithm to deal with 'ants' with dynamics characteristics (the ships). Several examples have been presented, showing satisfactory results. Althought they have been not included in this paper, the method has been also succesfully tested for longer distances but for these cases is always possible to divide the problem in two parts: navigation towards the objective and then manoeuvring to reach the goal.

At present, it is not a real time method, the algorithm it is not a light one, the CA is very time consuming for big environments, but sometimes the first solutions are available in a reasonable time –we are talking about minutes–. From the point of view of an offline method, it could be possible to systematically generate a library of solutions for routinary manoeuvers. This library could be stored and used by an onboard ship control system.

In preparation of future work, we have studied some scenarios with obstacles, obtaining encouraging results. Practical cases with obtacles are harbours and shallow waters. Another line of research is the improvement of the CA to make it auto-scalable depending on the scenario defined, to make it less time consuming.

REFERENCES

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. Oxford University Press, USA.

Dorigo, M. and St¨utzle, T. (2004). *Ant colony optimization*. MIT press.

Dormand, J. and Prince, P. (1980). A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6, 19–26.

Escario, J.B.e.a. (2008). Optimizaci´on de redes ad hoc mediante el algoritmo ant colony optimization. In *Proceedings of URSI-XXIII, Simposium Nacional de la Union Cientifica Internacional de Radio.*

Giron-Sierra, J., Cifuentes, S., and Jim´enez, J. (2006). Multi pseudo bang-bang control genetic optimisation for ship trajectory planning. In *Proceedings of IFAC International Conference on Manoeuvering and Control of Marine Craft.*

Gordon, D. (1999). *Ants at work: how an insect society is organized*. Free Press.

Ilachinski, A. (2001). *Cellular automata: A discrete universe*. World Scientific.

LaValle, S.M. (2006). *Planning Algorithms*. Cambridge University Press, Cambridge, U.K.

Lloyd, A. (1998). *Seakeeping: Ship Behaviour in Rough Weather*. A.R.J.M. Lloyds, Gosport, Hampshire, U.K.
P´olya, G. (1971). *How to solve it: A new aspect of mathematical method*. Princeton University Press Princeton, NJ.