Preprints of the 2012 20th Mediterranean Conference on Control & Automation (MED), Barcelona, Spain, July 3-6, 2012

Robot navigation in simulated pedestrian areas based on swarm intelligence

Jesús Espelosín* , Leopoldo Acosta and Alberto Hamilton

Abstract— **In this paper we present a method that relies on swarm intelligence to confront the problem of navigating in non-structured dynamic environments. The collective behavior of a group of particles is used to find an obstacle-free path that is able to adapt its course as the structure of the environment changes. Of particular importance is the rule that governs collision avoidance in that it not only takes into account the positions of the obstacles, but also their velocities.**

I. INTRODUCTION

One of the main problems in robotics is Robot Motion Planning. Through the years, two different approaches have been used. Classical methods such as Voronoi graphs, cell decomposition and potential fields, among others, form one group, with the other consisting of heuristic methods, such as those that feature prominently in the literature on Ant Colony Optimization [1], [2] and Genetic Algorithms [3], [4], [5].

Some heuristic methods are based on mimicking groups of animals that are particularly efficient at solving certain problems. The main idea of the method presented here is to use the simulated behavior of a flock of birds to solve the path planning problem in dynamic settings. Realistic results can be obtained by applying a different set of simple rules separately to each member of the flock, as was shown in [6]. Different behaviors are included as part of the flock's movement, such as obstacle avoidance, keeping a certain distance among neighbors or with respect to a specific point. Taking this into account, the members of the flock are used to explore the environment, searching for obstacle-free paths.

The positions of the flock's members make up a net of nodes that is then used as a search grid and, by applying the Dynamic Particle Chain (DPC) algorithm, yields a solution to the problem. In addition, the method is able to reconfigure the path when an obstacle intersects it, which makes it particularly well-suited to dealing with *complex* dynamic situations.

Some research has been done using the principles described by Reynolds to solve the problem of path planning [7], [8], [9], [10], [11], [12]. It is important to note that the method proposed is not an optimization method.

Manuscript received on January 30, 2012

Jesús Espelosín, *Corresponding author. Departamento de Ingeniería de Sistemas y Automática y Arquitectura y Tecnología de Computadores (ISAATC),Universidad de La Laguna, La Laguna 38203, Spain. jespelosin@isaatc.ull.es

Leopoldo Acosta. Departamento de Ingeniería de Sistemas y Automática y Arquitectura y Tecnología de Computadores (ISAATC),Universidad de La Laguna, La Laguna 38203, Spain leo@isaatc.ull.es

Alberto Hamilton. Departamento de Ingeniería de Sistemas y Automática y Arquitectura y Tecnología de Computadores (ISAATC),Universidad de La Laguna, La Laguna 38203, Spain alberto@isaatc.ull.es

This paper is organized as follows. The problem and necessary mathematical definitions are introduced in Section (II). The modifications to the rules described by Reynolds are explained in Section (III). Section (IV) details the operation of the simulator, as well as of the DPC method, and shows the graphical user interface that was programmed to test and adjust the algorithm.

II. THE METHOD

The data available in each iteration of the algorithm are the position of the target and the vehicle, as well as the position and the velocity of the moving obstacles in the environment. The information on the obstacles can be used to calculate the configuration space (CS), that is, the fraction of the space that is considered free of obstacles. A set of particles named boids is controlled by the algorithm, searching for a path from the vehicle to the target through the CS.

A. A path

In this study, the main aim that a path planning algorithm should accomplish can be summarized as trying to connect the vehicle's position with the target. In order to do this, the algorithm sequentially creates a set of connections between the vehicle's position, the particles distributed over the space, and the target. If there are no obstacles between the connections, the method creates a valid path and the algorithm is considered to have found a solution. Taking into account this explanation, the path is formed by a set of boid positions.

The flock is led toward the target while avoiding obstacles. This behavior allows the algorithm to focus its exploration on certain zones of the space that are likely to contain better paths while discarding other areas that are qualitatively less important.

The method features the following properties:

- It is able to explore many different paths simultaneously.
- When a complete solution is not available, the algorithm is able to generate partial solutions, such that the vehicle can advance toward the target even when the path is obstructed.

III. FLOCK BEHAVIOR DESCRIPTION

The dynamic of the flock is controlled by a set of rules. Each one has a specific purpose, so the individual effects must be combined in order to achieve the desired behavior.

978-1-4673-2529-5/12/$31.00 ©2012 IEEE

A. Proposed rules

In order to properly calculate the movement of each boid belonging to the flock, three vector variables must be stored: the position $\mathbf{p}_b(i)$, the velocity $\mathbf{v}_b(i)$ and the acceleration $\mathbf{a}_b(i)$. These vectors are defined by the following mathematical expressions:

$$\begin{aligned}
\mathbf{F}_b(i+1) &= \sum_{j=1}^{N_r} w_j \mathbf{R}_j \\
\mathbf{a}_b(i+1) &= m_b^{-1}\mathbf{F}_b(i+1) \\
\mathbf{v}_{aux}(i+1) &= \mathbf{v}_{Max}\frac{\mathbf{v}_b(i)+\mathbf{a}_b(i+1)}{|\mathbf{v}_b(i)+\mathbf{a}_b(i+1)|}; \\
\mathbf{v}_b(i+1) &= \mathrm{Min}\{\mathbf{v}_b(i+1)+\mathbf{a}_b(i+1), \mathbf{v}_{aux}(i+1)\} \\
\mathbf{p}_b(i+1) &= \mathbf{p}_b(i)+\mathbf{v}_b(i+1), \text{for } b \in \mathscr{I}_{N_b}. \qquad (1)
\end{aligned}$$

The vector sum of all the rules or steering behaviors is the total force $\mathbf{F}_b(i+1)$ applied over each boid of the flock. The state of the members of the flock, as well as the position and velocity of the obstacles along the environment and the position of the target, all influence the result of the rules.

In order to create a suitable behavior such that the flock finds paths free of obstacles, five steering behaviors were defined. These are explained below.

1) Alignment: The members of the flock are constantly aligning with their neighbors due to the effect of the alignment rule. The interesting effect of this steering behavior occurs when only part of the flock detects an obstacle. Those boids that, due to their distance from the obstacle, have not detected it are able to avoid it if they try to align with their flockmates. This rule thus generates an anticipative obstacle avoidance effect.

The particular form of this rule is

$$\mathbf{R}_{ali} = \frac{\sum_{k=1,k\neq b}^{N_b} g_k \mathbf{v}_k}{\sum_{k=1,k\neq b}^{N} g_k} - \mathbf{v}_b$$

with

$$g_k = \begin{cases} 1 & \text{if } ||\mathbf{p}_b - \mathbf{p}_k|| < T_{ali} \\ 0 & \text{otherwise.} \end{cases} \qquad (2)$$

Notice that this rule computes the mean velocity of a set of flockmates close to the reference boid and subtracts the velocity of the boid from said velocity. As a result of this steering behavior, the boid tends to align itself with its flockmates, as expected. Additionally, the local character of the rule is encoded in the parameter T_{ali} (visibility). In fact, locality with certain visibility is a general requirement for all rules governing flock dynamics.

2) Separation: So as to explore more of the environment, the flock members obey a separation rule. Each boid maintains a certain distance from its flockmates, avoiding collisions and allowing for different paths to be explored. A repulsion effect is generated from each boid in the form of a force generated over the other flockmates, propelling the boids of the flock towards the free space.

$$\mathbf{R}_{sep} = \frac{\sum_{k=1,k\neq b}^{N_b} g_k(\mathbf{p}_k - \mathbf{p}_b)}{\sum_{k=1,k\neq b}^{N} g_k ||\mathbf{p}_k - \mathbf{p}_b||^2} - \mathbf{v}_b$$

with

$$g_k = \begin{cases} 1 & \text{if } ||\mathbf{p}_b - \mathbf{p}_k|| < T_{sep} \\ 0 & \text{otherwise.} \end{cases} \qquad (3)$$

3) Cohesion: In order to make the collective behavior of the flock possible, a boid should not be too distant from its neighbors. This requires implementing a rule that allows the flock to remain cohesive. Specifically, the center of mass of the neighbors of the boid to which the rule is being applied is calculated. This center of mass exerts an attractive force such that the boid is steered towards its flockmates.

$$\mathbf{R}_{cohe} = \mathbf{h} - \mathbf{v}_b$$

with

$$\mathbf{h} = \begin{cases} \frac{\sum_{k=1,k\neq b}^{N_b} g_k \mathbf{p}_k}{\sum_{k=1,k\neq b}^{N} g_k} - \mathbf{p_b} & \text{if } ||\mathbf{p}_b - \mathbf{p}_l|| > T_{cohe} \\ (\mathbf{p_l} - \mathbf{p_b}) & \text{if } ||\mathbf{p}_b - \mathbf{p}_l|| < T_{cohe} \end{cases}$$

and

$$g_k = \begin{cases} 1 & \text{if } ||\mathbf{p}_b - \mathbf{p}_k|| < T_{cohe} \\ 0 & \text{otherwise} \end{cases}$$

4) Attraction towards the target: This rule is necessary to steer the flock towards the target. All members of the flock are attracted by the target with a force proportional to the distance between the boids and the target.

$$\mathbf{R_{target}} = \mathbf{p}_T - \mathbf{p}_b \qquad (4)$$

5) Obstacle avoidance: The most important restriction applicable to the members of the flock is obstacle avoidance. If this restriction is successfully implemented, the search for an obstacle-free path can proceed correctly. Taking into account the characteristics of a dynamic environment, this rule needs to know the position $\mathbf{p}_k$ and velocity $\mathbf{v}_k$ of the obstacles, as well as the position of the boid $\mathbf{p}_b$ and the target $\mathbf{p}_T$. The mathematical expression for this rule is presented below:

$$\mathbf{R}_{obs} = \frac{\sum_{k=1}^{N_O} g_k(\mathbf{p}_b - \mathbf{p}_k + \mathbf{c}_k)}{\sum_{k=1}^{N_O} g_k ||\mathbf{p}_b - \mathbf{p}_k||^2} - \mathbf{v}_b$$

$$\mathbf{c}_k = \begin{cases} \mathbf{l}_k & \text{if } (\mathbf{v}_k \triangleleft (\mathbf{p}_k - \mathbf{p}_b) \geq (\mathbf{v}_k \triangleleft (\mathbf{p}_T - \mathbf{p}_b)) \\ 0 & \text{otherwise} \end{cases},$$

$$\mathbf{l}_k = \begin{cases} \mathbf{v}_k & \text{if } ((\mathbf{p}_k - \mathbf{p}_b) \triangleleft (\mathbf{p}_T - \mathbf{p}_b)) > T_{avoid} \\ -\mathbf{v}_k & \text{if } ((\mathbf{p}_k - \mathbf{p}_b) \triangleleft (\mathbf{p}_T - \mathbf{p}_b)) \leq T_{avoid} \end{cases}$$

$$\text{and} \qquad g_k = \begin{cases} 1 & \text{if } ||\mathbf{p}_b - \mathbf{p}_k|| < T_{obs} \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathbf{a} \triangleleft \mathbf{b}$ denotes the angle formed by the vectors $\mathbf{a}$ and $\mathbf{b}$.

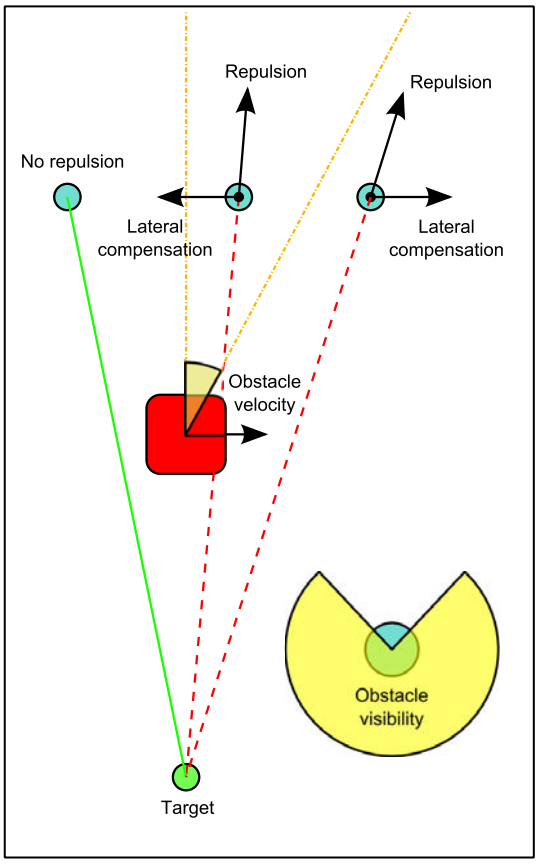

Fig. 1. Obstacle avoidance rule (cf. discussion in sec. III-A.5).

Only the obstacles 135 degrees to either side of the front affect the boid. The main idea of this rule is to generate a behavior that provides a certain anticipatory effect, such that every boid in the flock can effectively avoid moving obstacles. Three different possible cases are used to define the problem.

- When the future trajectories of the boid and the obstacle do not intersect. In this case the obstacle does not affect the boid's movement.
- When the future trajectories of the boid and the obstacle intersect. This means that a collision danger exists that requires changing the direction of the boid. If the intersection condition is satisfied, a repulsive force is generated from the obstacle to the boid. Two further distinctions can be made:
  - The obstacle is not located within the front angle threshold. In order to help the boid avoid the obstacle, a lateral compensation equal to the obstacle velocity is added to the repulsive force.
  - The obstacle is located within the front angle threshold. In this case the lateral compensation has an equal but opposite magnitude to the obstacle velocity.

To clarify these concepts, please refer to Figure (1).

IV. Trajectory Generation

A graphical user interface (GUI) was designed to display the progress of the algorithm in real time. A set of controls was added to vary the value of the different parameters in the method. The main window shows the flock, the target, the vehicle and the flock, as well as the calculated path.

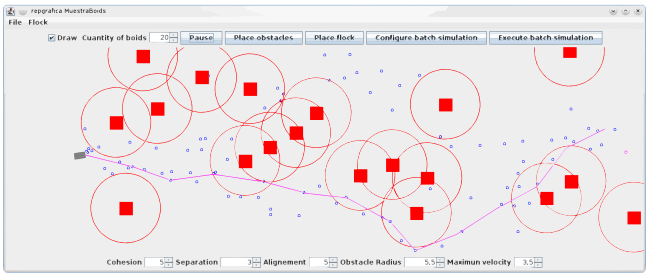

Fig. 2. Graphical user interface (cf. discussion in sec.IV).

In each iteration of the algorithm, the simulator carries out five different steps:

1) At the beginning, the positions and velocities of the obstacles are updated.
2) The rules are calculated for each boid in the flock and its position is updated. This is how the flock is moved.
3) Path generation. The distribution of the boids over the scenery is used to create an obstacle-free path. How this path is constructed will be detailed in the next paragraph.
4) Path tracking. The path geometry is used to generate the control commands that steer the vehicle along the environment . The steer command is calculated by obtaining the angular difference between the vehicle's yaw and the line that links the second point of the trajectory and the vehicle. The first point of the trajectory is the vehicle position. As for the velocity, the vehicle tries to maintain the maximum velocity allowed as long as no objects are within the security threshold. If an obstacle is nearby, the reduction in the vehicle's speed is inversely proportional to its distance from the obstacle.

5) In order to terminate the simulation, the vehicle must be within a certain pre-defined distance from the target. If this condition is not satisfied, steps 1 to 4 are repeated.

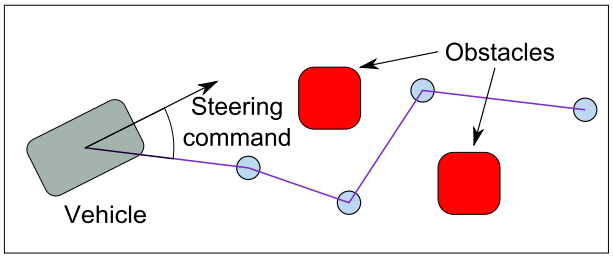

Fig. 3. Steering command. (cf. discussion in sec.IV).

In order to generate a correct path without overloading the computing resources available, the vehicle adds boids into the scenery as a source of particles. Inversely, boids that are close enough to the target are removed. A diagram with these simulator steps is shown in Figure (4).

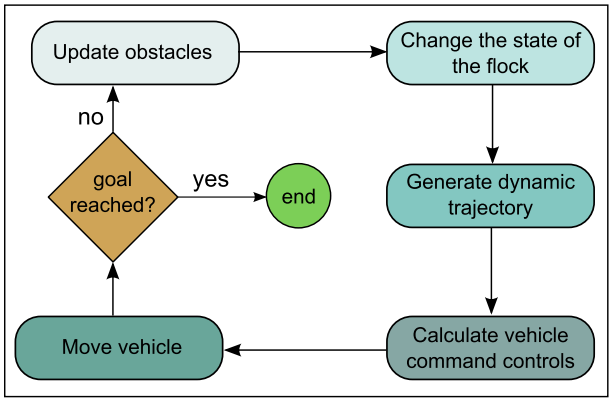

Fig. 4. Main algorithm. (cf. discussion in sec.IV).

The main goal of the DPC strategy is to find an obstacle-free path that connects the vehicle's position to the target. This path is created by making connections among the positions of the boids over the space. A graphical explanation of the algorithm is shown in Figure (5). As noted previously, the position of the vehicle is assigned as the first point on the trajectory. In order for a boid in the flock to be selected, three conditions have to be satisfied. The candidates to be joined to the trajectory have to be within a proximity radius T_{search} around the previous point. The boid that is closest to the target is selected, but only if there are no obstacles between the previous point and the candidate boid.

It is possible that no boid will be present within the proximity radius that complies with the three aforementioned conditions. In this case, the boids belonging to the trajectory are counted. If the result of this count is zero, this means that there are no available paths for this iteration and the vehicle stops temporarily. If, on the other hand, the trajectory has at least one boid, then a partial valid path is considered to exist, so the vehicle does not stop and continues crossing the environment. If a valid boid exists near the previous point, it is added to the path. If the last point added to the trajectory is close enough to the target, a complete path is considered to have been created and the search process in the present iteration of the simulator ends. If this is not the case, the process continues adding boids to the trajectory until the positions of the vehicle and the target are connected.

To demonstrate the capabilities of this algorithm, we designed a 100-m wide by 40-m long test area. Depending on the difficulty level, it is possible to vary the number of obstacles, the positions and velocities of which are randomly determined. When the experiment starts, the velocity of the obstacles is kept constant. The vehicle's maximum velocity is limited to 3m/s and its steering angle to +/- 30 degrees. The Ackerman model was used to simulate the behavior of the vehicle. Both the traction motor and the automatic steering system were also modeled.

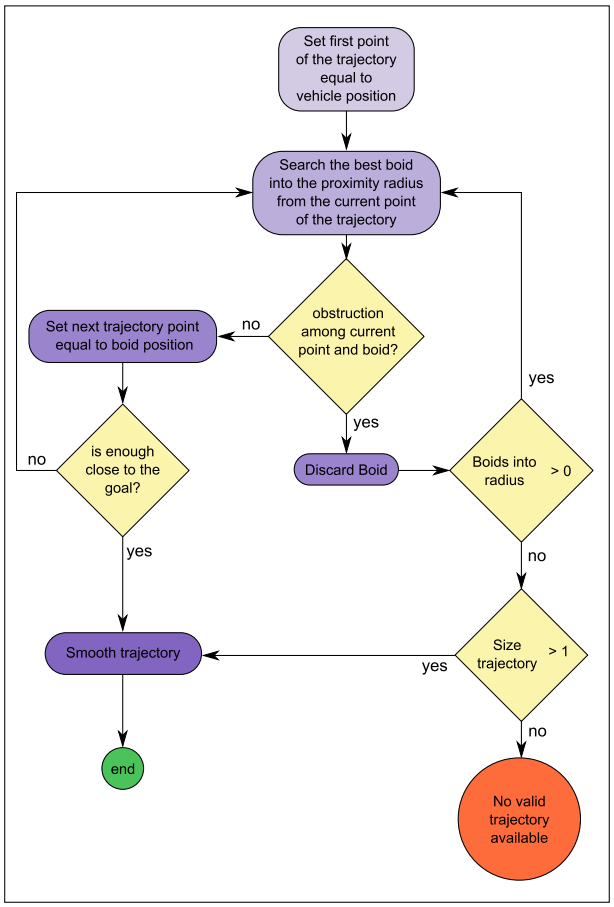

Fig. 5. (Dynamic trajectory algorithm cf. discussion in sec.IV).

V. CONCLUSIONS AND FUTURE WORK

A. Conclusions

The combined action of the set of rules proposed in this research was designed so as to solve complex dynamic situations. The positions of the boids are adapted in order to find a valid pseudo-optimal path. One notable characteristic of the algorithm is that the method is able to generate a

partial path even when the path to the target is obstructed. By taking into account the dynamic characteristics of the environment, the method is designed to find a complete trajectory to the target.

Also, thanks to the obstacle avoidance rule, the algorithm has the ability to anticipate future collisions with obstacles. The flock is steered towards future free space, which makes finding clear trajectories possible. This is yet another remarkable feature worth noting in light of the particular difficulties posed by environments with moving obstacles.

B. Future Work

Since the simulation uses a model of the vehicle that correctly represents its dynamics, and considering the adequate performance shown by the DPC algorithm, it should be possible to carry out a live implementation of the method. In fact, our initial motivation was precisely this, to develop an actual prototype that will be able to deal with the problem of path planning in pedestrian areas.

VI. ACKNOWLEDGMENTS

This research was supported by the project SAGENIA DPI 2010-18349, and by the Spanish Ministry of Education and Science, project FIS2010/19998. J. E.'s work is funded by the Agencia Canaria de Investigación, Innovación y Sociedad de la Información (ACIISI)

REFERENCES

[1] M. Dorigo, T. Stützle, The Ant Colony Optimization Metaheuristic: Algorithms, Applications and Advances, in: Handbook of Metaheuristics, 2002, pp. 250–285.

[2] M. Dorigo, T. Stützle, Ant Colony Optimization, MIT Press: Cambridge, MA, USA, 2004.

[3] H. Chen, Z. Xu, Path planning based on a new genetic algorithm, in: International Conference on Neural Networks and Brain, IEEExplore, vol 2, 2005, pp. 788–792.

[4] M. Gemeinder, M. Gerke, An active search algorithm extending ga based path planning for mobile robot systems, Soft Computing and Industry, Springer, Berlin (2002) 589–596.

[5] R. Haupt, S. Haupt, Practical Genetic Algorithms, second edi Edition, Wiley, NJ, United States, 2004.

[6] C. W. Reynolds, Flocks, herds and schools: A distributed behavioral model, ACM SIGGRAPH Computer Graphics 21 (4) (1987) 25–34. doi:10.1145/37402.37406.

[7] P. Raja, S. Pugazhenthi, Path Planning for Mobile Robots in Dynamic Environments Using Particle Swarm Optimization, 2009 International Conference on Advances in Recent Technologies in Communication and Computing (2009) 401–405doi:10.1109/ARTCom.2009.24.

[8] Q. Ma, X. Lei, Q. Zhang, Mobile Robot Path Planning with Complex Constraints Based on the Second-Order Oscillating Particle Swarm Optimization Algorithm, 2009 WRI World Congress on Computer Science and Information Engineering (2009) 244–248doi:10.1109/CSIE.2009.124.

[9] E. Masehian, D. Sedighizadeh, A multi-objective PSO-based algorithm for robot path planning, 2010 IEEE International Conference on Industrial Technology (2010) 465–470doi:10.1109/ICIT.2010.5472755.

[10] A. Z. Nasrollahy, H. H. S. Javadi, Using Particle Swarm Optimization for Robot Path Planning in Dynamic Environments with Moving Obstacles and Target, 2009 Third UKSim European Symposium on Computer Modeling and Simulation (3) (2009) 60–65. doi:10.1109/EMS.2009.67.

[11] C. Shi, Y. Bu, J. Liu, Mobile robot path planning in three-dimensional environment based on ACO-PSO hybrid algorithm, 2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (2008) 252–256doi:10.1109/AIM.2008.4601668.

[12] S.-h. Kim, G. Lee, I. Hong, J. Kim, D. Kim, New potential functions for multi robot path planning : SWARM or SPREAD, 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE) (2010) 557–561doi:10.1109/ICCAE.2010.5451658.