An improved Tangent Bug method integrated with artificial potential field for multi-robot path planning

Conference Paper · June 2011

DOI: 10.1109/INISTA.2011.5946136

CITATIONS
19

READS
1,213

3 authors, including:

Khaled El-Metwally
Cairo University
45 PUBLICATIONS **840** CITATIONS

SEE PROFILE

An Improved Tangent Bug Method Integrated with Artificial Potential Field for Multi-robot Path Planning

Emam Fathy Mohamed
Arab Academy for Science and Technology
Heliopolis, Cairo 2033 Egypt
emfmz@cairo.aast.edu

Khaled El-Metwally
Cairo University
Dept. of Electrical Power Eng.
Giza, 12211Egypt
Kmetwally@eng.cu.edu.eg

A.R. Hanafy
Cairo University
Dept. of Electrical power Eng.
Giza, 12211Egypt
arhanafy@eng.cu.edu.eg

Abstract—**This paper introduces the utilization of the wall following concept for path planning of multi-robots and proposes an improved Tangent Bug method to avoid falling in local minima encountered by the artificial potential field (APF) method used in real-time path planning. In the new algorithm, more reliable switching and merging conditions are designed to guarantee the success of escape. This method controls a team of robots with a deformable geometry, compliant with nearby static or dynamic obstacles including those represented by each robot team-mates. The robots are virtually linked to each other by the influence of artificial potentials that asymptotically stabilize the formation and keep all the robots separated by specified distances. Simulation studies have been carried out to verify the validity of the proposed method.**

Keywords- tangent bug; potential field; local minima; obstacle avoidance; multi-robot systems.

I. INTRODUCTION

Autonomous navigation of a robot team relies on the ability of the team to achieve their goal, avoiding the obstacles in the environment. In some cases the team has a complete knowledge of its environment, and plans its movement based on it. But, in general, the team only has an idea about the goal, and should reach it using the robots sensors to gather information about the environment.

Hierarchical systems decompose the control process by function. Low-level processes provide simple functions that are grouped together by higher-level processes in order to provide overall vehicle control [1].

One can think of high level processing as planning level where high level plans are generated, and low level processing as reactive control where each robot needs to avoid the obstacle sensed by its sensors. This is called local path planning.

Local path planning should be performed in real time, and it takes priority over the high level plans. Therefore, it is some time called real time obstacle avoidance.

One of the local path planning methods, is the potential field method [2]. It is an attractive method because of its elegance and simplicity [3]. However, using this method the robot can be easily fall in a local minimum and some other problems. Therefore, additional efforts are needed to avoid this situation.

This paper is organized as the following: section 2 introduces the potential field method. Then section 3 introduces some limitations with potential field method. Section 4 describes methods to avoid the local minimum. Section 5 presents a brief survey about Bug algorithms. Section 6 presents our improved tangent bug algorithm. Section 7 discusses our implementation of the algorithm. Section 8 describes the experiments and the obtained results. Finally, the paper concludes.

II. ARTIFICIAL POTENTIAL FIELDS

Artificial Potential Fields [2] is an iterative technique commonly adopted in the area of robotics for controlling the navigation of robots in dynamic environments. Robots are modeled as particles moving in a field of potentials attracted by low potential areas; the position to be reached generates an attractive force (a low potential zone) and obstacles generate repulsive forces (high potential zones). At each iteration the particle moves along the force resulting from the sum of all repulsive (obstacle avoidance) and attractive (goals) forces influencing current particle position; the particle continues to move until it reaches a stable state [4]. Figure1 shows a robot navigation system affected by different forces.

Potential field is defined as the sum of an "attractive" potential U_{att} and a "repulsive" potential U_{rep}:

The potential is given by:

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{1}$$

With the following resulting force:

$$F(q) = -\nabla U_{att}(q) - \nabla U_{rep}(q) \tag{2}$$

Where $U_{att}(q)$ is the attractive potential that pulls the robot towards the goal and $U_{rep}(q)$ is the repulsive potential that causes the robot to avoid obstacles.

The attractive potential and its gradient are defined as function of the actual configuration q and the goal configuration q_{goal}:

$$U_{att}(q)=\begin{cases}\frac{1}{2}\zeta d^2(q,q_{goal}), & d(q,q_{goal})\le d^*_{goal}\\ d^*_{goal}\zeta d(q,q_{goal})-\frac{1}{2}\zeta(d^*_{goal})^2, & d(q,q_{goal})\rangle d^*_{goal}\end{cases} \quad (3)$$

$$\nabla U_{att}(q)=\begin{cases}\zeta(q-q_{goal}), & d(q,q_{goal})\le d^*_{goal}\\ \frac{d^*_{goal}\zeta(q-q_{goal})}{d(q,q_{goal})}, & d(q,q_{goal})\rangle d^*_{goal}\end{cases} \quad (4)$$

Where ζ is the attractive scaling factor, d^*_{goal} is the attractive distance threshold; both factors are selected by trial and error.

The repulsive potential and its gradient are defined as a function of the distance D(q) between the actual configuration and the closest obstacle:

$$U_{rep}(q)=\begin{cases}\frac{1}{2}\eta(\frac{1}{D(q)}-\frac{1}{Q^*})^2, & D(q)\le D^*\\ 0, & D(q)\rangle D^*\end{cases} \quad (5)$$

$$\nabla U_{rep}(q)=\begin{cases}\eta(\frac{1}{Q^*}-\frac{1}{D(q)})\frac{1}{D^2(q)}\nabla D(q), & D(q)\le D^*\\ 0, & D(q)\rangle D^*\end{cases} \quad (6)$$

Where η is the repulsive scalar, and Q^* is the repulsive distance threshold; both factors are selected by trial and error.

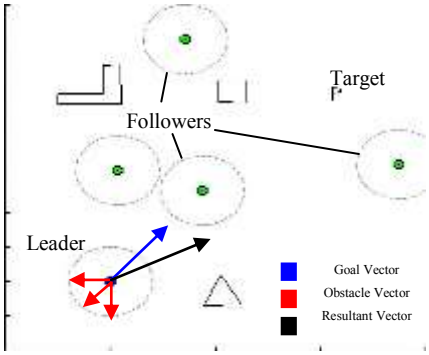

Figure 1. Robot navigation system.

The paper adopts the implementation of this algorithm since it is very useful for real purposes, although it has some known problems.

III. LOCAL MINIMA

Potential field method is attractive because of its mathematical elegance and simplicity. However, several limitations inherent in potential field method had been addressed systematically based on mathematical analysis done in [5]. This paper deals with one famous problem which is local minima.

Originally proposed and well-suited for on-line planning where obstacles are sensed during motion execution, the potential field method present a well known problem associated with the possible existence of local minima situations, that can get robots blocked and trapped in certain configurations. This occurs very frequently behind obstacles where repulsive forces counterbalance the attraction forces.

To deploy the robot team to a target area, a local minima free path will be needed to guide the leader throughout the obstacle field. The Interaction Forces between all the robots will drag the robot formation behind and in some situations; robots can get stuck in a local minima configuration if the leader (blue) is drawn away and around an obstacle. Figure 2 shows an example of this situation with the leader reaching the goal and a team member is getting stuck in a local minimum while trying to follow the leader.

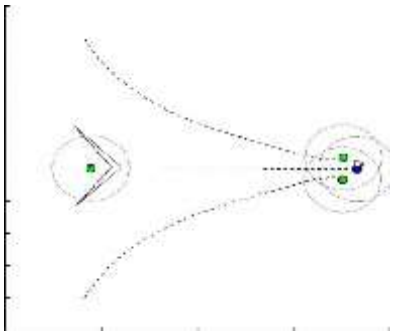

Figure 2. One team member is getting stuck in a local minimum while trying to follow the leader.

IV. AVOIDING THE LOCAL MINIMUM

Several methods have been suggested to deal with the local minimum phenomenon in potential filed method. One idea is to avoid the local minimum by incorporating the potential field with the high-level planner, so that the robot can use the information derived from its sensor, but still plan globally [1].

Another set of approaches are to allow the robot to go into local minimum state, but then try to fix this situation by:

1) Backtracking from the local Minimum and then using another strategy to avoid the local minimum.
2) Doing some random movements, with the hope that these movements will help escaping the local minimum.
3) Using more complex potential fields that are guaranteed to be local minimum free, like harmonic potential fields [1].
4) Changing the potential field properties of the position of the local minimum. So that if the robot gets repelled from it gradually.
5) Using a procedural planner, such as wall following, or using one of the bug algorithms to avoid the obstacle where the local Minimum is.

All these approaches rely on the fact that the robot can discover that it is trapped, which is also an ill-defined problem.

The method used in this work, is similar to the last point, where we integrate a modified Tangent Bug algorithm with a Potential Field method to overcome the local minima.

V. BUG ALGORITHMS

All Bug type algorithms are based on the fact that the shortest distance between two points is a straight line. Therefore, the efforts to find the path for a mobile robot are such that to make it as close as possible to the straight line that crosses the start and the end point. When this cannot be obtained as, for example, an environment with obstacles, the algorithms tries first to contour the obstacles and then it resumes the path by following the Origin- Destiny line [6].

In the Bug2 algorithm, the robot starts moving directly toward the destination. When an obstacle is found, the robot begins contouring it, only resuming moving to the destiny point when a certain leaving condition is met. In this method the leaving condition consists on finding the Origin-Destiny line while contouring an obstacle.

A. Tangent Bug

It serves as an improvement to the Bug2 algorithm in that it determines a shorter path to the goal using a range sensor with a 360 degree infinite orientation resolution. The robot can move more efficiently toward the goal also go along shortcuts when contouring obstacles and switch back to goal seeking earlier [7].

Tangent Bug iterates between two behaviors: motion-to-goal and boundary-following. These behaviors are different than in the Bug1 and Bug2 approaches. Although motion-to-goal directs the robot to the goal, this behavior may have a phase where the robot follows the boundary. Likewise, the boundary-following behavior may have a phase where the robot does not follow the boundary.

In the motion-to-target mode, the robot moves in a straight line towards the goal or towards a vertex of an obstacle for which the path through that vertex is the shortest know path. If the robot has to move away from the goal, the boundary-following mode is initiated.

When the robot follows an obstacle boundary, the algorithm uses the local tangent graph to do so. While moving around the obstacle's boundary, the value (minimum distance) d_{min} is updated to be the shortest distance between any point on the obstacle and the goal. The robot stops moving along an obstacle's boundary once it finds a point on the local tangent graph which is closer to the goal than d_{min}. If such a point is found, the robot moves to this point and resumes the motion to target mode once the robot is closer to the goal than d_{min}.

Figure 3 shows a robot which is controlled by the Tangent Bug algorithm. The dotted circles show the sensor range of the robot. In this figure, one can see how the robot initially moves towards the goal. Once the first obstacle in its path comes within sensor range, the robot avoids it and moves towards the edge of it. Once free from the obstacle it moves directly towards the goal again until the process is repeated with the second obstacle. When compared with the other Bug algorithms, the Tangent Bug algorithm gives a path which is much closer to the optimal path.

Tangent Bug algorithm is designed to work in environments with stationary obstacles. A new algorithm which shall be referred to as the Improved Tangent Bug algorithm (ITB) is developed to optimize the Tangent Bug algorithm for dynamic environments. The ITB algorithm is discussed in the following Section.

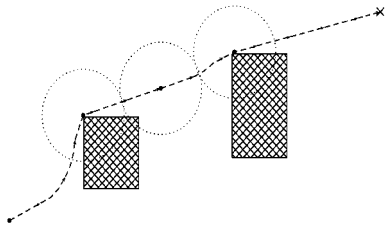

Figure 3. A robot guided by the Tangent Bug algorithm.

B. Improved Tangent Bug Algorithm

The ITB algorithm developed as a part of this paper to adapt the Tangent Bug algorithm for a dynamic environment and to improve robots formation algorithm in [8]. Like the Tangent Bug algorithm, the ITB algorithm also uses two modes, namely, the motion-to-goal mode and the boundary-following mode. The most notable difference is that the ITB algorithm does not need to observe all the obstacles around it.

In the motion-to-goal mode, if an obstacle is blocking the robot's path to the goal, the ITB algorithm only looks as far as the left and right edges of the obstacle. Once both edges of the blocking obstacle have been found, the robot moves past the edge which gives the shortest path to the goal. As in the Tangent Bug algorithm, if the next position moves the robot away from the goal, the boundary-following mode is initiated.

In the boundary-following mode, the ITB algorithm will note the shortest distance between any point on the obstacle it is following and the goal. Before moving around the obstacle, the algorithm first checks if there is any point to move to which is closer to the goal than any point on the obstacle. If there is, the robot moves towards this point and the boundary-following mode is terminated. If no such point is visible, the robot continues to move around the obstacle in the direction of the last motion-to-goal movement.

VI. ALGORITHM

Our planning algorithm classifies the robot's behavior into two modes, one is to go directly to the target from the current position using potential field (behaving as a team member also using potential field) [8]. The other mode is to use IBT algorithm to avoid static obstacles (once reached a previously specified distance from the obstacle).

Whilst using ITB algorithm, two variables are continuously updated:

1) $d_{followed}$ - is the shortest distance between the sensed boundary and the goal.
2) d_{reach} - is the shortest distance between blocking obstacle and goal (or my distance to goal if no blocking obstacle visible).

The ITB mode persists (or switched on) until one of the following switching conditions occur:

1) $d_{reach} < d_{followed}$.
2) The robot has encircled the minimum-causing obstacle (the target is unreachable).

The pseudo-code of the algorithm is given in this section.

Algorithm 1: Path Planning Simulation Program

Data: *set of Leaders, set of goals, set of each leader followers.*
Result: *new set of robots follow their leaders*

```
while TerminationCondition = false do
    for each Leader do
        Draw Leaders new position
        Draw Goal
        for robots of each leader do
            Draw Robots
    Draw Obstacles
    for each Leader do
        Calculate distances between Leaders and their goals
        Calculate distances between Leaders "DL"
        Calculate distances between Leaders and static
              obstacles "LS"
        if (LS > safe distance)then
            Use Potential field mode for leaders
            for robots of each leader do
                Use Potential field mode
            for each Leader do
                for robots of each leader do
                    Calculate distances between robots of a leader
                    and the other robots "DRR"
                    if (DRR < safe distance)then
                        Calculate Potential between robots of a leader
                        and the other robots
            else
                Use ITB algorithm
                for robots of each leader do
                    Use Potential field mode
                for each Leader do
                    for robots of each leader do
                        Calculate distances between robots of a leader
                        and the other robots "DRR"
                        if (DRR < safe distance)then
                            Calculate Potential between robots of a leader
                            and the other robots
        if (DRL for all teams < specified config. distance) & at
            goal then
            TerminationCondition = true
```

VII. EXPERIMENTS

To illustrate our approach more explicitly and to show the robustness of the algorithm, a set of experiments performed in a simulated environment to test and evaluate the algorithm presented in this paper. With simulation, multiple scenarios can be presented, that may also include uncommon and rare situations difficult to model and verify in a more practical environment. The advantages do not come without cost because simulation errors exist, and practical implementation is, by itself, a whole new problem that may even render the theoretical project unpractical. The risk becomes even more present when strong assumptions are made like, for example, assuming that each robot knows his exact position and of all his team-mates, the obstacle detection will function perfectly, and no problems will arise due to some other aspects like, for example, friction, slippage, acceleration and velocity limits, etc. But even with these limitations, the presented simulated results help bring a better understanding of the algorithms and opens way for future practical experimentations. The set of simulations presented in this part are performed using Matlab.

A. Single robot

In the first experiment, figure 4 shows how a single robot uses ITB algorithm alternately with potential field method to reach the goal, depending on - the previously mentioned-switching conditions.

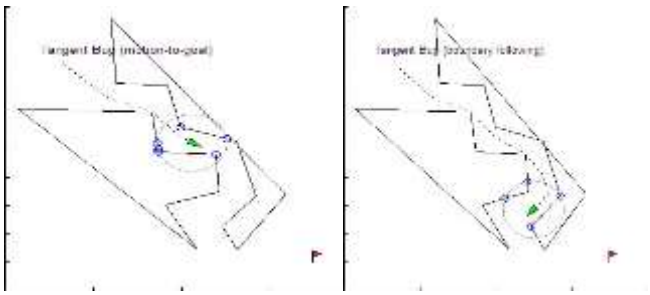

Figure 4-a. Tangent Bug (motion to goal) mode (left); then Tangent Bug (boundary following) mode (right).

Figure 4-b. Potential Field mode again and reaching the Target.

B. Multi-robots without coordination

In figure 5, multi-robots are trying to reach different goals. Each robot is using a different mode inside the algorithm at the same time, as indicated in the description of the figures.

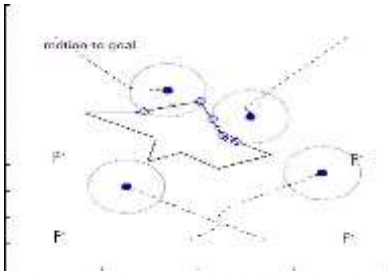

Figure 5-b. Two robots are using potential field mode only and the others are using ITB mode to avoid the static obstacle.

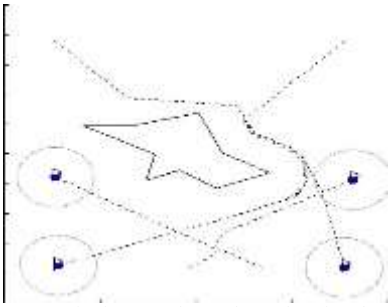

Figure 5-c. Final Scene shows the paths of each robot.

C. *Multi-robot with coordination*

In figure 6, the algorithm is tested for multi-robots-multi-teams. The followers (green) use only the potential filed mode; while the leaders (blue) use the potential field algorithm alternately with the ITB algorithm according to the switching conditions.

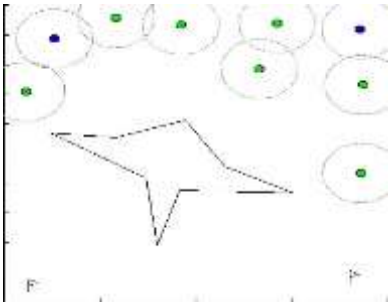

Figure 6-a. Each team leader and member is set at a starting position.

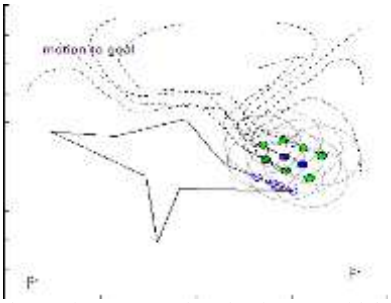

Figure 6-b. Each team is constructed; and trying to avoid the obstacles and reach the targets.

Figure 6-c. Formations reached targets.

CONCLUSION

In this research, an improved Tangent Bug algorithm is introduces to avoid falling in local minima in potential field based path planning. Switching conditions have been designed in the algorithm, where the robot needs to decide which mode should be switched to. Simulation studies show that the improved Tangent Bug method overcomes some drawbacks of potential field method. Moreover, it is suitable for multi-robot path planning as shown in simulation.

However, for the practical application of the proposed method, experiments on actual robots are necessary and the further improvement should be made for guaranteeing the ability of the method to overcome more problems in potential field. On the other hand, not only existing in the APF method, the local minimum problem is prevalent in many other reactive navigation approaches, e.g., the fuzzy logic control methods [9], and [10]. Hence the wall following concept proposed in this paper may be improved for more general applications.

REFERENCES

[1] Gregory Dudek, Michael Jenkin, "*Computational principles of mobile robotics*", Cambridge University Press, Cambridge, 2000,ISBN: 0-521-56021-7.

[2] O. Khatib, "Real-time Obstacle Avoidance for Manipulators and Mobile Robots", *Proceedings of the IEEE International Conference on Robotics & Automation*, 1985, pp.500-50.

[3] Ding Fu-guang, Jiao Peng, Bian Xin-qian, Wang Hong-jian, "AUV local path planning based on virtual potential field", Mechatronics and Automation. *IEEE International Conference*, 2005, Vol.4, Pages: 1711-1716.

[4] P. Burelli, and A. Jhala, "Dynamic Artificial Potential Fields for Autonomous Camera Control". *Proceedings of the Fifth Artificial Intelligence in Interactive Digital Entertainment Conference (AIIDE09)*, Stanford, CA. 2009.

[5] Y. Koren and J. Borenstein, "Potential field methods and their inherent limitations for mobile robot navigation". in *Proc. International Conference on Robotics and Automation*, Sacramento, California. 1991.

[6] R. Siegwart and I. R. Nourbakhsh, Introduction to Autonomous Mobile Robots, MIT Press, 2004.

[7] I. Kamon, E. Rimon, and E. Rivlin. Tangentbug, "A range-sensor based navigation algorithm*". Int. Journal of Robotics Research*, 1998, 17(9):934–953.

[8] Emam Fathy, Khaled El-Metwally, A. R. Hanafy, "Multi-Robot Tracking of Multiple Moving Targets Using Potential Field Approach". *International Symposium on Innovations in Intelligent Systems and Applications*, Turkey, 2010, Pages: 471- 475.

[9] C. Ordonez, E. G. Collins Jr., M. F. Selekwa, D. D. Dunlap. (2008). The virtual wall approach to limit cycle avoidance for unmanned ground vehicles. *Robotics and Autonomous Systems*, vol. 56, pp. 645–657.

[10] O. R. E. Motlagh, T. S. Hong, N. Ismail, "Development of a new minimum avoidance system for a behavior-based mobile robot". *Fuzzy Sets and Systems,* 2008, vol. 160, pp. 1929-1946.