POTENTIAL FIELD MULTI-OBJECTIVE OPTIMIZATION FOR ROBOT PATH PLANNING USING GENETIC ALGORITHM

HUSSEIN HAMDY SHEHATA*, JOSEF SCHLATTMANN

Systems Technology and Design Methodology, Hamburg University of Technology
21073 Hamburg, Germany

Path planning and autonomous navigation algorithms play a vital role in the field of robotics. Amongst these, the potential field algorithm is widely used due to its elegant mathematical model. Although it serves the basic purpose of avoiding obstacles, it is bounded by particular restrictions. The use of a virtual obstacle along with potential field algorithm is a lucrative approach to overcome these limitations. This work aims at optimizing certain parameters involved in the virtual obstacle concept by the use of Non-Dominated Sorting Genetic Algorithm II (NSGA II). It is advisable to maintain a safety margin around the obstacle and to maneuver efficiently without oscillations as it moves close to the obstacle. Furthermore, the size of the robot also affects its motion. This paper takes into account all these factors during the optimization process. The results have proven its feasibility and validity in unknown environments.

1. Introduction

Robotics has been an exciting field for researchers for a very long time. Advancements in technology and the need for more automation in daily life lead to new and increasing challenges in this field. The robot should plan a path that enables it to avoid collision with the obstacles. Amongst many algorithms, the potential field algorithm is widely used. The Artificial Potential Field (APF) method is analogous to a ball rolling downhill. It is first proposed by Khatib [1] for manipulators and mobile robots. Koren and Borenstein [2] described the virtual force field concept and the limitations were identified [2]. An improved wall following behavior coupled with potential field was proposed by Zhu, Zhang, and Song [3]. Jia and Wang [4] represented obstacles as a set of linear segments along their boundary and defined the repulsive potential as a line integral along its contour. Calculating linear integral increases the computational cost during online path planning. Velagic, Lacevic, and Osmic [5] combined the APF method with an obstacle pruning method that used the concept of visibility field to solve the problem of Goals Not Reachable due to Obstacles Nearby (GNRON). Yin and Yin [6] modified the attractive potential by including the

* Corresponding author: Tel.: +49(0)40-42878 4430; E-mail: shehata@tu-harburg.de

relative acceleration in the goal potential field. Li, Cui, and Lu [7] also included the goal distance in the obstacle potential. However, the repulsive potential was modified to change exponentially with the distance from the obstacle. The exponential decrease of potential around the obstacle can however cause the paths to be very close to obstacle raising safety issues. To tackle this situation, a virtual obstacle concept for obstacle avoidance was proposed by Shehata and Schlattmann [8]. A robot size factor was introduced in the virtual obstacle potential field. The work described in [8] was further extended for motion planning in dynamic environments [9].

2. Motivations and Objectives

Many researchers have aimed at modifying the APF to overcome the limitations faced in the conventional APF. Several works have included the concept of a safe distance around the obstacles by introducing parameters which can be varied to allow different levels of clearance. But very few of them have actually calibrated the respective parameters to allow different safety margins depending upon the application. The current work aims at optimized path planning using artificial potential field and virtual obstacle method described in [8]. For safety of the robot in real time implementation, it is advisable to maintain a minimum distance around the obstacle and to maneuver efficiently without oscillations. Several test cases are used to demonstrate the validity of the optimized parameter values in different static scenarios using MATLAB.

3. Multi-Objective Optimization using NSGA-II

3.1. *Background*

The Classical APF method is a gradient descent method used for robot path planning. The obstacles in the workplace are represented by a repulsive potential field. The goal is represented by an attractive potential field. The potential function can be considered as energy and hence the negative gradient of the potential function is the force vector pointing in the direction of decreasing potential. For a robot located at $q_r = [x_r\ y_r]^T$, a goal located at $q_g = [x_g\ y_g]^T$ and the obstacles are located at $q_{oi} = [x_{oi}\ y_{oi}]^T$, where i = 1,2,3,..,n and n denotes the number of obstacles. The attractive force $F_{att}(q_r)$ and the repulsive force $f_{repi}(q_r)$ from the i^{th} obstacle are given by (1) and (2), respectively.

$$F_{att}(q_r) = \zeta(q_g - q_r) \tag{1}$$

$$f_{rep_i}(q_r) = \begin{cases} \eta\left(\dfrac{1}{\rho(q_r, q_{o_i})} - \dfrac{1}{\rho_o}\right) \\ \left(\dfrac{1}{\rho^2(q_r, q_{o_i})}\right)\nabla\rho(q_r, q_{o_i}), & \text{if } \rho(q_r, q_{o_i}) \le \rho_o \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where ζ and η are positive scaling factors, $\rho(q_r, q_{oi})$ is the minimum distance from the robot to the i^{th} obstacle, ρ_o is a positive constant denoting the distance of influence of the obstacle, and $\nabla\rho(q_r, q_{oi})$ is a unit vector pointing from the i^{th} obstacle to the robot. The virtual obstacle concept was proposed in [8]. The force from this virtual obstacle is similar to that of a real obstacle but altered by a positive scaling factor, λ. The force due to this virtual obstacle is as follows:

$$f_{vo}(q_r) = \begin{cases} \eta\left(\dfrac{1}{\lambda}\cdot\dfrac{1}{\rho(q_r, q_{vo})} - \dfrac{1}{\rho_o}\right) \\ \left(\dfrac{1}{\lambda^2}\cdot\dfrac{1}{\rho^2(q_r, q_{vo})}\right)\nabla\rho(q_r, q_{vo}), & \text{if } \rho(q_r, q_{vo}) \le \rho_o \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where, $f_{vo}(q_r)$ denotes the repulsive force from the virtual obstacle, $\rho(q_r, q_{vo})$ is the minimum distance from the robot to the virtual obstacle, and $\nabla\rho(q_r, q_{vo})$ is a unit vector pointing from the virtual obstacle to the robot. It can be seen from (3), as λ increases, $f_{vo}(q_r)$ decreases and vice versa. The direction of the virtual obstacle is located along the extension line from the goal to the robot and in the opposite side of the goal. This location prevents the robot from falling into local minima or oscillations anymore and enhances the movement toward the goal. The position of virtual obstacle is given by:

$$\rho(q_r, q_{vo}) = \frac{\rho(q_r, q_g) + min\left(\rho(q_r, q_{o_i})\right)}{2} \tag{4}$$

where $\rho(q_r, q_g)$ and $min(\rho(q_r, q_{oi})$ are the distances from the robot to the goal and to the closest obstacle, respectively.

For the rest of this work, the following assumptions are considered: 1) maximum distance ρ_o up to which the obstacle potential acts on the robot is 2m, and 2) maximum allowable step size for a robot = the radius of the robot, R_r. The reader has to notice that the maximum allowable step size refers to the maximum travel per each instant.

3.2. *Optimization of the Robot Size Factor, λ*

Consider a 2D workspace with a point robot located at $q_r = [0.5\ 1]^T$, an obstacle at $q_o = [1.5\ 1]^T$ and the goal (target) located at $q_g = [2\ 1]^T$ as shown in Figure 1. As can be seen, when λ = 0.5, the robot follows a longer detour around the obstacle while for λ = 0.3, it moves very close to the obstacle. The value of λ can thus be adjusted depending on the size of the robot and the required clearance of the path from the obstacle. For a very small value of λ, the repulsive force from virtual obstacle dominates significantly over the normal obstacle force causing the robot to move close to the obstacle. A suitable value of λ is therefore a compromise between safety of robot and the length of path.

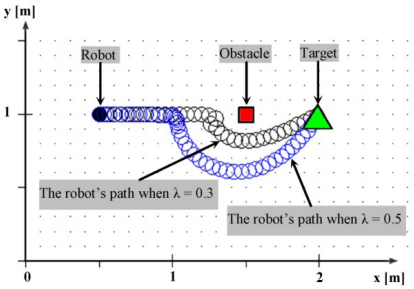


Figure 1. The influence of the robot size factor on path's length

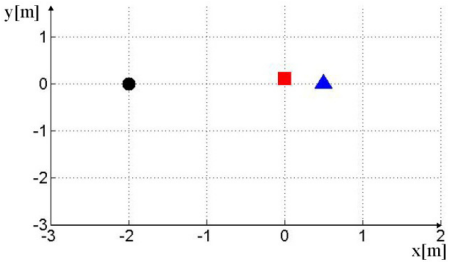


Figure 2. A simple test case for a static environment

Considering the drawbacks in classical potential theory, placing the goal very close to the obstacle would be a better option for the test case. All these conditions can be tested during the robot run in the test case shown in Figure 2. To avoid a local minima situation due to the robot, the obstacle and the goal are all collinear; the obstacle is placed at a small offset of 0.1m. For a given robot size, λ should maneuver the robot at the desired safety margin and avoid oscillations in the robot motion. A fitness function is created for the paths that take into account both these conditions. The fitness function comprises of 2 objectives:

❶**Distance:** This objective helps to maintain a safety margin around the robot. A variable *safe* is calculated at each position of the robot path:

$$safe = \rho(q_r, q_{o_i}) - R_r - safety\ margin \tag{5}$$

Safe < 0 implies the robot is within the safety margin. *Safe* = 0 implies the robot is exactly at a safety margin. *Safe* > 0 implies the robot is outside the safety margin. Therefore, for all positions of the robot for which *safe* < 0, a high penalty *w* is placed on it. That means if *safe* < 0, then *safe* = *safe* **w*.

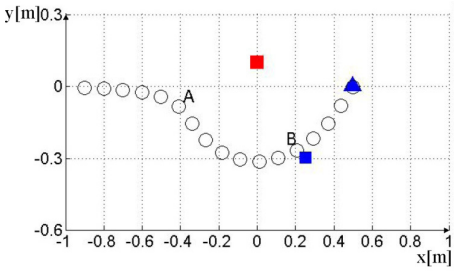


Figure 3. The robot's path for the test case with λ = 0.64; step size = 0.1m

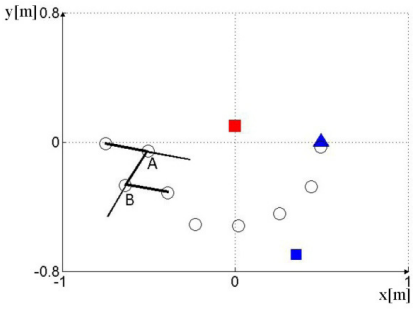


Figure 4. Test case with λ = 0.8; step size = 0.25m

Figure 3 shows enlarged view near the goal of the path of a robot with diameter 0.4m. $\lambda = 0.64$ and step size = 0.1m. The value *safe* is calculated only for the robot positions from A to B. Before position A, the robot is at a distance larger than the required clearance. After position B, it is at a distance larger than the radius and moving away from the obstacle. Since we wish to minimize the fitness function, all paths very close to the obstacle and very far from it should have higher values. Paths which are at a distance nearly equal to the safety margin should have lower values.

$$d = \sum_{path} \left(\frac{safe}{S}\right)^2 \tag{6}$$

where S is a normalizing factor. Since the obstacle potential acts to a maximum distance of 2m, $S = 2$.

❷**Angle:** This parameter is used to minimize oscillations. We consider the angle between the two consecutive path segments, e.g. A and B as shown in Figure 4. It shows the enlarged view as the robot of diameter 0.5m approaches the obstacle. The value is maintained within $\pm\pi$ radians. We formulate the parameter *alphapath* as in (7):

$$alphapath = \sum_{path} \left(\frac{astep}{\pi}\right)^2 \tag{7}$$

where *astep* is the angle between any two consecutive path segments. Since *astep* corresponds to the change in the direction of the robot between two consecutive steps; a turn with $\pm\pi$ radians is acceptable. Therefore, *astep* is normalized by π as shown in (7).

Fitness function: The fitness function considered here is as follows:

$$F = w_1.alphapath + w_2.d \tag{8}$$

where w_1 and w_2 are the weights assigned to *alphapath* and *d*, respectively. The weighting functions w_1, w_2 and w can be varied for different robot sizes to obtain a good minima condition. After several trials, it was observed that $w_1 = w_2 = 1$ and $w = 20$ gives optimal values of λ. The test case is run for values of λ which varies in the range [0, 1]. *d* and *alphapath* have to be calculated for each robot path according to the value of λ. Figure 5 shows the plots of the objectives *d* and *alphapath*, the fitness function and the test cases.

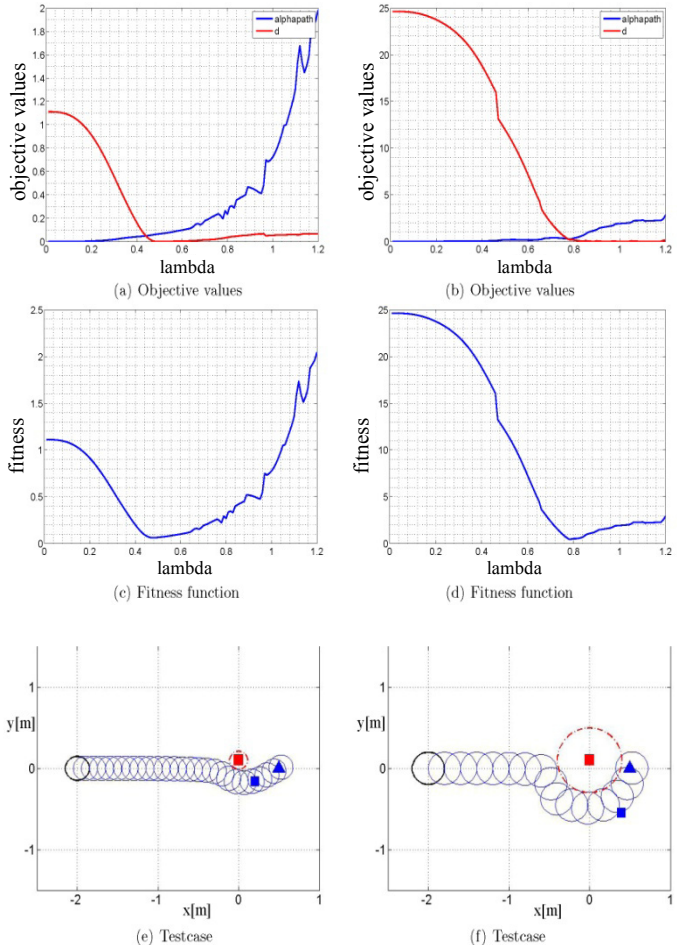
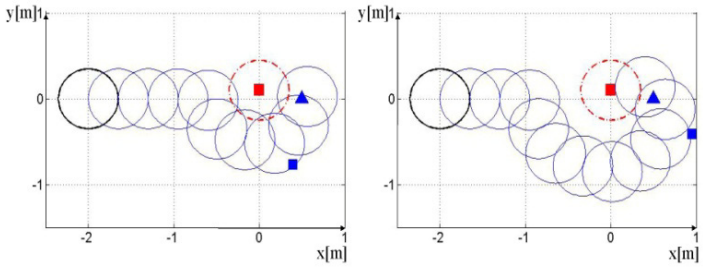


Figure 5. Plots of objectives (top), fitness function (middle) and test case paths (bottom) for robot diameter 0.3m (left) and 0.4m (right)

3.3. *Optimization of λ and η*

Figure 6 (a) shows the test case for the robot diameter 0.7m moving at a step size of 0.35m at a safety margin = radius =0.35m. The optimal lambda value generated is 0.8. It can be seen that the robot enters significantly within the safety margin. Since λ affects the force from virtual obstacle which points in the direction of the goal, optimizing λ alone would imply optimizing the push towards the obstacle to shape the path. On the other hand, η is a multiplying factor in the obstacle potential and hence the repulsive force. Therefore, if λ and η are both simultaneously modified, it should create a push-pull effect on the robot generating a safer and smoother path. The objectives and the fitness function are kept the same. The test case is run for all combinations of λ varying in (0, 1) and η in (1, 6). Again we attempt to minimize the fitness function which will correspond to the ideal values of λ and η. On running this algorithm for the same test case, the values obtained are λ = 0.6 and η = 5.15. Figure 6 (b) shows the path generated using these values. While the length of the path has been increased in comparison to the earlier run, it is a safe path. In Figure 7 (a), the robot of diameter 0.8m is moving with a step size of 0.3m maintaining a safety margin = radius = 0.4m. Figure 7 (b) shows the test run for a 0.5m diameter robot moving at 0.1m step size.



(a) Optimizing λ (b) Optimizing λ and η

Figure 6. Test case for robot diameter 0.7m with a safety margin of 1*radius

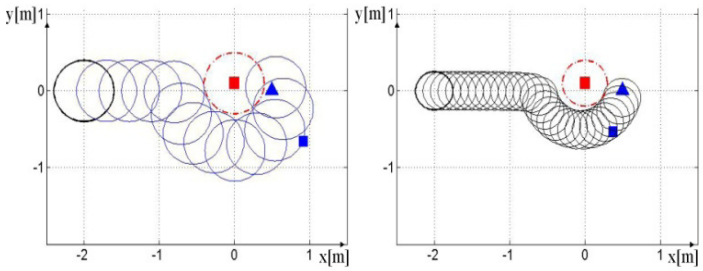


Figure 7. Test case for different parameters combinations

3.4. *Optimization of λ, η and Step Size*

It would be more convenient if the objectives are simultaneously optimized for different values of step size. We only need to find optimal values of λ and η for varying step sizes. Objective d is kept the same as in (6) except for $w = 3$. For objective *alphapath*, the value *astep* is calculated as described in section 3.2 with the following modification:

$$if\,|astep| > \frac{\pi}{2}; \quad then\ astep = 5.|astep| \tag{9}$$

where |*astep*| is the absolute value of *astep* and the weighting factor 5 is set intuitively to impose a high penalty on turning angle greater than 90^o. An additional penalty is imposed on *astep*:

$$if\ safe < 0.04; \quad then\ astep = |astep| + 100\pi.\frac{(-0.04 - safe)}{safety\ margin} \tag{10}$$

A very small value of λ will result in a nearly straight line path with large d values and small *astep* at each step along the path. Equation (10) ensures that these steps in the path that lie more than 0.04m inside the safety margin are penalized to have high *astep* and therefore high *alphapath* values given by:

$$alphapath = \sum_{path} \left|\frac{astep}{\pi}\right| \tag{11}$$

Consider a robot of radius = 0.35m moving at a step size of 0.7*radius. Allowing η values = 1, 3, 5, 7 and 9 and $\lambda \in (0, 1)$ (at intervals of 0.1), the values of d and *alphapath* are calculated for the test case and plotted in Figure 8. The safety margin = 0.75*radius. A desired value of λ and η would minimize d and *alphapath* and would therefore lie in the lower left corner of the plot. Points along the Pareto front imply a trade off between the objectives. The Non-dominated Sorting Genetic Algorithm II (NSGA-II) described in [10] can be used to identify this Pareto front. If the step size is also varied in the plot of Figure 8, we get a 3D graph as shown in Figure 9. NSGA-II is explained in algorithm 1.

Algorithm 1: NSGA-II optimization of λ and *η* for different step sizes

Input: robot diameter, maximum allowable step size, clearance margin, crossover and mutation rates, population size (N) and maximum number of generations.

Output: set of combinations of λ, *η* and step sizes

1. Generate a random population of size N with $\lambda \in (0, 1)$, $\eta \in (0, 6)$ and step size varying discretely from 0.1*radius to 1*radius. This is the parent population for the first generation.
2. Run the test case for each member of the parent population and calculate *d* and *alphapath* for each member.

3. Perform non-dominated sorting of the population.
4. Determine crowding of the members based on step size and η density.
5. Select N pairs of parents from the population through roulette and binary selection based on non-dominated ranks and step size and η crowding.
6. Perform crossover and mutation on each pair. Repeat the process till a new population of size N is generated.
7. Merge the new and parent population together forming a total size of 2N.
8. Repeat step 2 and step 3 for the new population of size 2N.
9. Select N parents to produce the next generation based on non-dominated rank and crowding.
10. Generate a new set of individuals by performing step 5 and 6 on these N parents.
11. If the number of generations has exceeded the maximum number of generations, then stop. Else repeat the steps 7-11.
12. Repeat the steps 6-8 on the last generated population.

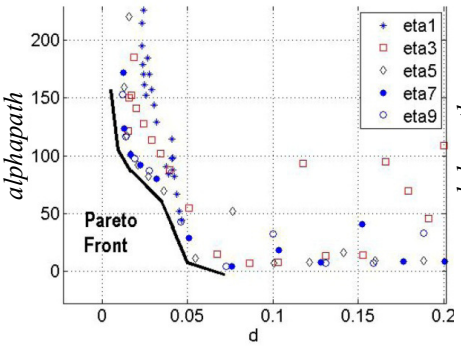


Figure 8. Enlarged view showing Pareto front

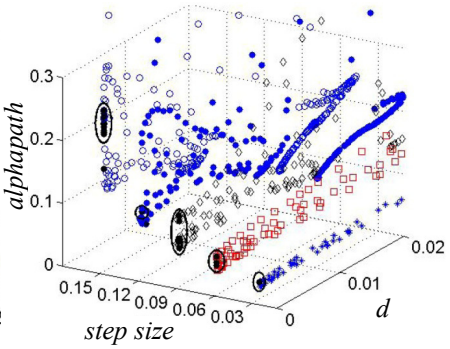


Figure 9. 3D plot of *d*, *alphapath* and step size for robot diameter 0.6m

4. Conclusion

The optimization of the parameters involved in the potential functions is carried out to ensure a safe path around the obstacles in a workspace. In case of static environment, the parameter λ of virtual obstacle is first optimized. It is observed that the optimization of a single parameter does not always succeed in keeping a safe distance from the robot. As a result, η and λ are simultaneously optimized. The resulting values obtained improve the safety of the robot around obstacles. However, a robot does not move at a constant step size during its motion. For identifying the values of the two parameters at different step sizes, the procedure needs to be repeated multiple times. To avoid this, the Non Dominated Sorting Algorithm II (NSGA II) is used to obtain the values of λ and η for multiple discrete step sizes. The combination of parameters thus obtained enables the robot in the test case to move around the obstacle while staying at the desired

distance from the obstacle. This provides a method of dynamically changing the force field around the robot without any heavy computational cost incurred during real time motion.

References

1. O. Khatib, Real-Time Obstacle Avoidance for Manipulators and Mobile Robots, *International Journal of Robotics Research*, **5**, 90 (1986).
2. Y. Koren and J. Borenstein, Potential Field Methods and their Inherent Limitations for Mobile Robot Navigation, in *Proceedings of the IEEE Conference on Robotics and Automation*, Vol.2, pp.1398–1404, Sacramento, California (1991).
3. Y. Zhu, T. Zhang, and J. Song, An Improved Wall Following Method for Escaping from Local Minimum in Artificial Potential Field Based Path Planning, in *Proc. of the 48th IEEE Conference on Decision and Control,* pp. 6017–6022, (2009).
4. Q. Jia and X. Wang, Path Planning for Mobile Robots Based on A Modified Potential Model, in *Proc. of the International Conference on Mechatronics and Automation,* pp. 4947–4952, (2009).
5. J. Velagic, B. Lacevic, and N. Osmic, Efficient Path Planning Algorithm for Mobile Robot Navigation with A Local Minima Problem Solving, in *Proc. of the International Conference on Industrial Technology*, pp. 2325–2330, (2006).
6. L. Yin and Y. Yin, An Improved Potential Field Method for Mobile Robot Path Planning in Dynamic Environments, in *the 7th IEEE World Congress on Intelligent Control and Automation*, pp. 4847–4852, (2008).
7. C. Li, G. Cui, and H. Lu, The Design of An Obstacle Avoiding Trajectory in Unknown Environment Using Potential Fields, in *Proc. of the International Conference on Information and Automation,* pp. 2050–2054, (2010.
8. H. H. Shehata and J. Schlattmann, Mobile Robot Path Planning and Obstacle Avoidance Based on a Virtual Obstacle Concept, in *Proceedings of the 21st International Conference on Flexible Automation and Intelligent Manufacturing*, Vol. 2, No. 1, pp. 905–914, Taichung, Taiwan (2011).
9. H. Shehata and J. Schlattmann, Reactive Algorithm for Mobile Robot Path Planning Among Moving Target/Obstacles by Means of Dynamic Virtual Obstacle Concept, in *Proc. of the 22nd International Conference on Flexible Automation and Intelligent Manufacturing,* pp. 563–574, (2012).
10. K. P. Deb, A. S. Agarwal, and T. Meyarivan, A Fast Elitist Multiobjective Genetic Algorithm: Nsga-II, *IEEE Transactions on Evolutionary Computation*, **6**, 182 (2000).