PSO-Based Path Planning Algorithm for Humanoid Robots Considering Safety

Roham Shakiba*, Mostafa E. Salehi

Mechatronics Research Lab., Electrical, Computer, and IT Dept, Qazvin Branch, Islamic Azad University, Qazvin, Iran

Received 4 April 2012; accepted 8 August 2013

Abstract

In this paper we introduce an improvement in the path planning algorithm for the humanoid soccer playing robot which uses Ferguson splines and PSO (Particle Swarm Optimization). The objective of the algorithm is to find a path through other playing robots to the ball, which should be as short as possible and also safe enough. Ferguson splines create preliminary paths using random generated parameters. The random parameters are then iteratively fed into the PSO for optimization and converging to optimal path. Our proposed method makes a balance between the path shortness and the safety which makes it more efficient for humanoid soccer playing robots and also for any other crowded environment with various moving obstacles. Experimental results show that our proposed algorithm converges in at most 60 iterations with the average accuracy of 92% and the maximum path length overhead of 14% for planning the shortest and yet safest path.

Keywords: Path Planning, Ferguson Splines, Particle Swarm Optimization (PSO)

1. Introduction

Path planning is an important problem in the artificial intelligence fields which are based on mobile agents. Generally, the problem is finding the minimum length obstacle-free path for moving from one point to another autonomously. Other constraints such as execution time are also considered for its application. This problem covers a large variety of applications such as the moving robot arm for grabbing doorknob, navigating an autonomous plane in air, moving simulated agents in 2D or 3D, moving a patrolling robot in a planar field and so on.

Evolutionary algorithms are widely used in this field because path planning can be assumed as a constraint optimization problem. Some of the evolutionary algorithms such as ant colony or genetic algorithm achieved good results [1] but could not satisfy low storage, low computational complexity, and real-time constraints for applications such as soccer. PSO is a bio-inspired evolutionary algorithm which was proposed in 1995 by Kennedy and Eberhart [2] and in comparison with the other algorithms have benefits such as less candidate solutions and parameters, global optimization capability, and fast convergence. Many researchers have tried to use PSO for path planning such as the work in [3]; however, these algorithms cannot satisfy the output path smoothness which is required for human-like motion. While in [4], a stochastic PSO-based path planning with high-order polynomial is introduced for planning a smooth path, the high complexity of particle for the large number of parameters reduces the algorithm efficiency. In [5, 6], the combination of cubic Ferguson Splines with PSO is introduced, where traditional PSO algorithms with the premature convergence problem [7] is used. In [8], the introduced algorithm resolved the premature convergence problem using Clerc's PSO algorithm but there is no balance between path length and safety. This aims to planning a path through obstacles in cases that obstacles are close to each other and this is a drawback especially in applications like humanoid soccer play in which walking stability is a challenge by itself. So our presented method plans a curve path considering obstacles density and path length with respect to their importance and produces a normalized multi-objective fitness function.

* Corresponding author. Email: shakiba@qiau.ac.ir

2. Algorithm Components

In this section we introduce our proposed method.

2.1. Ferguson Splines

Ferguson spline is defined by the following equation [9]:

$$r(t) = P_0 f_1(t) + P_1 f_2(t) + P_0' f_3(t) + P_1' f_4(t) \quad (1)$$

Where P_0 and P_1 are the starting and end point coordinates and P_0' and P_1' are their corresponding tangent vector and t is the spline resolution vector parameter which belongs to [0,1]. Also $f_1(t)$ to $f_4(t)$ are Ferguson cubic multinomials described by:

$$\begin{aligned} f_1(t) &= 2t^3 - 3t^2 + 1 \\ f_2(t) &= -2t^3 + 3t^2 \\ f_3(t) &= t(t-1)^2 \\ f_4(t) &= t^2(t-1) \end{aligned} \quad (2)$$

According to this definition, if in successive splines, start and end points and their corresponding tangent vectors are equal, the resulting path will be smooth because: $r(0) = P_0$, $r(1) = P_1$ and also $r'(0) = P_0'$, $r'(1) = P_1'$. Our problem space is 2D so formula (1) and these steps must be applied for the x and y dimensions separately. Then $r_x(t)$ and $r_y(t)$forms a path in the 2D space. Finally the particle format which will be sent to the next step is shown in table 1.

Table 4

Particle structure which Feeds to PSO.

P_{0x}	P_{0y}	P'_{0x}	P'_{0y}	P_{1x}	P_{1y}	P'_{1x}	P'_{1y}	…	P'_{nx}	P'_{ny}

2.2. Particle Swarm Optimization (PSO)

PSO was introduced by Kennedy and Eberhart in 1995. It is inspired from social behaviour of human beings and can be used for finding global optima on some arbitrary functions. In PSO each problem solution is called particle and the collection of particles is called swarm. Each particle has a position $\overrightarrow{x_i}(t) = (x_{i1}, x_{i2}, \dots, x_{iD})$ and a velocity $\overrightarrow{v_i}(t) = (v_{i1}, v_{i2}, \dots, v_{iD})$ in d-dimensional problem space. Also every particle knows its best previous position $\overrightarrow{pBest_i}(t) = (p_{i1}, p_{i2}, \dots, p_{iD})$ and swarm global best position$\overrightarrow{gBest_i}(t) = (g_{i1}, g_{i2}, \dots, g_{iD})$. So a velocity function according to the best positions updates the particles speed and position till the swarm converges to the optimum position.

Here we used Clerc's PSO with velocity function:

$$\overrightarrow{v_i}(t+1) = \chi\left(\overrightarrow{v_i}(t) + c_1 r_{1i}\left(\overrightarrow{pBest_i} - \overrightarrow{x_i}(t)\right) + c_2 r_{2i}\left(\overrightarrow{gBest_i} - \overrightarrow{x_i}(t)\right)\right)$$

Where:

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|}, \varphi = c_1 + c_2, \varphi > 4$$

And c_1 and c_2 are random positive numbers which control the relative attraction to global and local best found positions. And r_1 and r_2 are vectors of random variables drawn with uniform probability from[0,1]. So in each iterationposition updates by: $\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \overrightarrow{v_i}(t+1)$. If any element v_{id} of velocity vector is less than V_{min} or greater than V_{max} it will be replaced with a randomly generated value in the above distance. Finally the most important part is the fitness function which evaluates the eligibilityof each path. Here we use a multi-objective fitness function which consists of two fitness functions:

2.2.1. Length Fitness

This fitness function must penalize long paths. So the function is:

$$f_l = \frac{L}{L_{min}},$$

Where $L_{min}$is the length of straight line which connects starting and end points and L is the trajectory length.

2.2.2. Safety Fitness

This fitness function must penalize trajectories with respect to their distance to obstacles considering obstacles' density. The obstacle aggregation in some applications (such as humanoid soccer) is important due to the following reasons:

- Motion stability is still not completely reliable. So neighbour robots are more dangerous.
- Respects to different robots sizes, it is possible that a small fast robot stick between multi big slow ones.
- Being surrounded between some huge robots will threat robot perception because robot localization is performed by detecting important spots in the field, therefore, no detection means losing the position in the field.

Accordingly we propose this function:

$$f_s = \begin{cases} 1 & , \nexists i \in O: 0 \le d_{min_i} < D_{Safe} \\ \sum_{i \in O} k_i e^{\frac{D_{Safe}+1}{d_{min_i}+1}-1} & , \exists i \in O: 0 \le d_{min_i} < D_{Safe} \end{cases}$$

Where the variable D_{Safe} is the safety margin constant which is 0.5^m according to the RoboCup rules. O is the set of all obstacles. d_{min_i}, is the minimum distance to the i^{th}obstacle. And k_i is the obstacle crowd coefficient which is defined by:

$$k_i = \begin{cases} 1 , & d_{min_i} > D_{Safe} \\ n(O_n), & 0 \le d_{min_i} < D_{Safe} \end{cases}$$

Where $n(O_n)$ is the number of obstacles which are located near this obstacle (inside $2D_{Safe}$ radius). This means if an obstacle is far enough k_i will be 1 otherwise a crowd coefficient will be applied to the exponential distance related function.

Finally fitness function is: $f = \frac{f_l}{k_{acc}+1} + \frac{k_{acc} f_s}{k_{acc}+1}$

Where $k_{acc} = \sum_{i \in O_n} k_i$. These coefficients insure that the two fitness function to be normalized.

3. Experimental Results Analysis

We have implemented this algorithm using MATLAB with these parameters:$V_{max} = 5, V_{min} = -5$, Maximum iteration number = 60, Swarm size = 20.

We used two fixed locations for the starting and target points (robot and ball) with 2 to 4 random obstacles located in random positions inside a $1^m \times 1^m$ area between endpoints. Then each of the random generated configurations will be fed into both the algorithm introduced in [8] and our new proposed method. It will be noticed that in the first step random parameters will shape trajectories which for a precise comparison, these random parameters are the same in both algorithms. These steps are repeated 10 times and the paths' lengths for both methods are recorded. Fig. 1 shows an output of our implemented algorithm. The workspace consists of 3 obstacles. The left column shows the method described in [8] and the right one is our proposed method. The figures at the top show workspace and the bottom ones show convergence of fitness function. Fig. 2 shows the difference between the two methods clearly. Here path length in our method is 22% longer than [8]. However, our method's path is safer. Fig. 3 demonstrates a workspace with 4 obstacles. Although the outcome of our proposed method (the right side) is the safest path in the 10th repetition, comparing its length with the average length indicates that different repetitions plan different paths (upside or downside of whole obstacles).The results for the first 25 workspaces (discarding cases which obstacles are so far, or too close) can be classified as the followings:

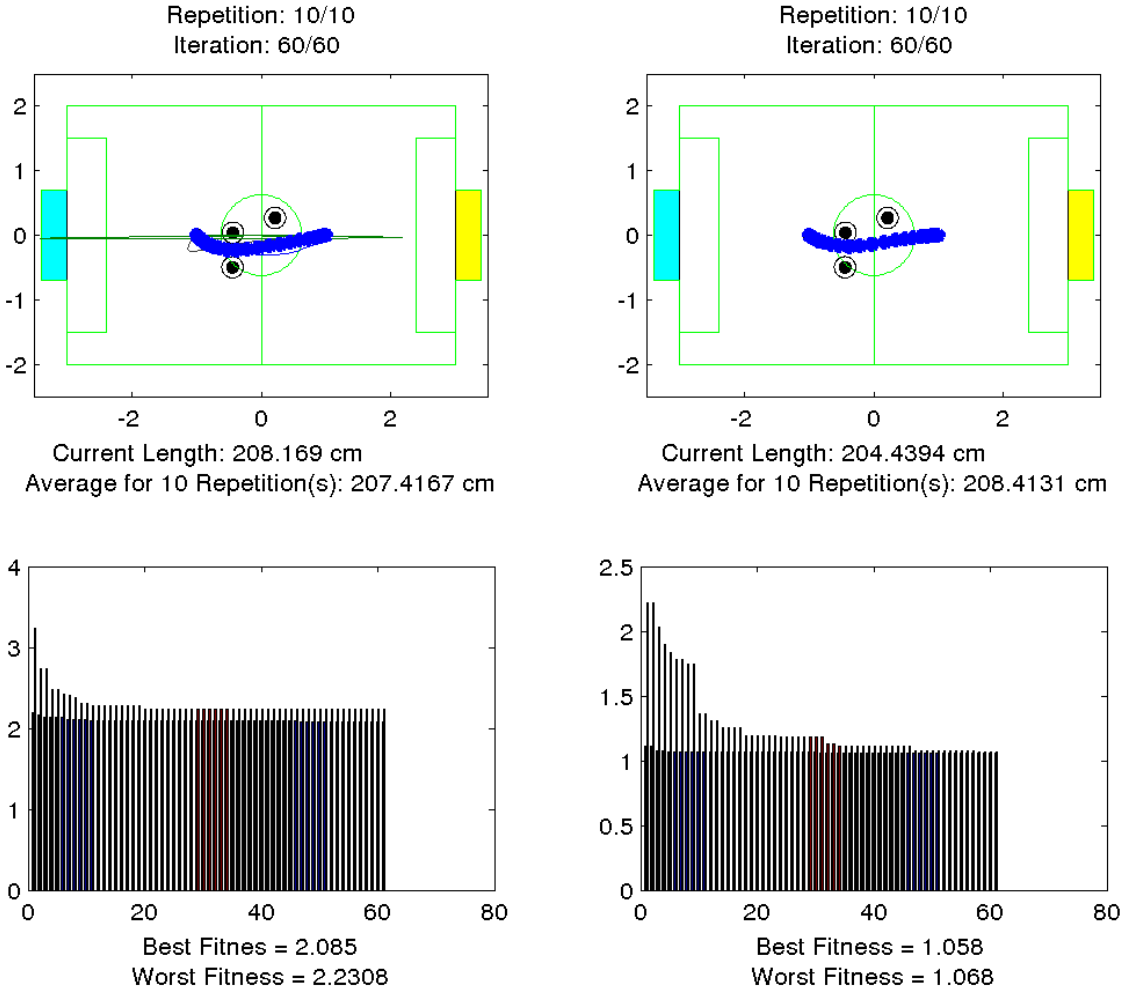

Fig. 1. An output of our algorithm with the same output as [8].

- In 60% of cases (such as Fig. 1) both methods are similar in which there is no significant difference in the planned path in the two methods. In these cases obstacles are far enough to pass through or very close as there is no way through. It is noticed that the best solution's fitness in our method is 1, which indicates normalized and balanced effect of length and safety in fitness function.
- In 32% of cases (see Fig. 2) the new method will return a path which is in average about 14% longer, but is safer. In these cases obstacles are

near each other (in $2D_{Safe}$ distance which is allowed for passing through according to the previous method)

- In the remaining 8% cases, different paths (with different length) are planned in different repetitions. In average minimum and maximum lengths overhead are about 11% and 25%,respectively which corresponds to planning safer and safest path (see Fig. 3).

According to the aforementioned analysis in 92% of the configurations, our new proposed method plans a path considering both length and safety. This means we select the shortest path if it is safe enough or the safest path with a maximum of 14% overhead in the path length. And in the remaining 8% of cases it is possible to plan safer or safest path with average length overhead of 11% or 25% respectively.

Also for investigating the criteria which causes planning different paths (in 8% of aforementioned random configurations) we studied two series of special workspaces each consisting of two fixed end points with a distance of 2^m and a set of obstacles with the same distance from the straight line connecting two end points. Initially obstacles are located far enough, in which our algorithm plans the straight line as output. Then obstacles are getting closer in each step until the straight line becomes unsafe. These space configurations and corresponding results are explained as follows.

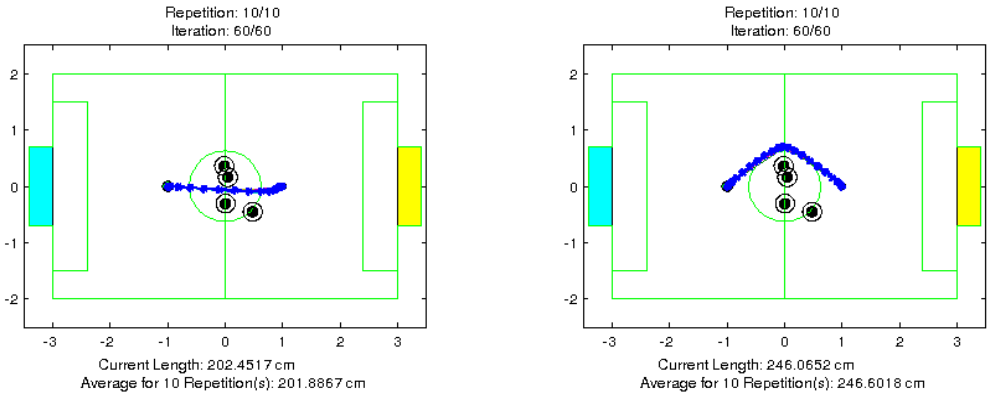

Fig. 2. This is a configuration with 4 obstacles. This figure shows the difference between the two methods in which our proposed method (right side) plans longer path with about 22% length overhead, while the previous method (left side) plans shorter unsafe path. Previous and proposed methods path lengths are $\mathbf{202}^{\mathbf{cm}}$ and $\mathbf{246}^{\mathbf{cm}}$ respectively.

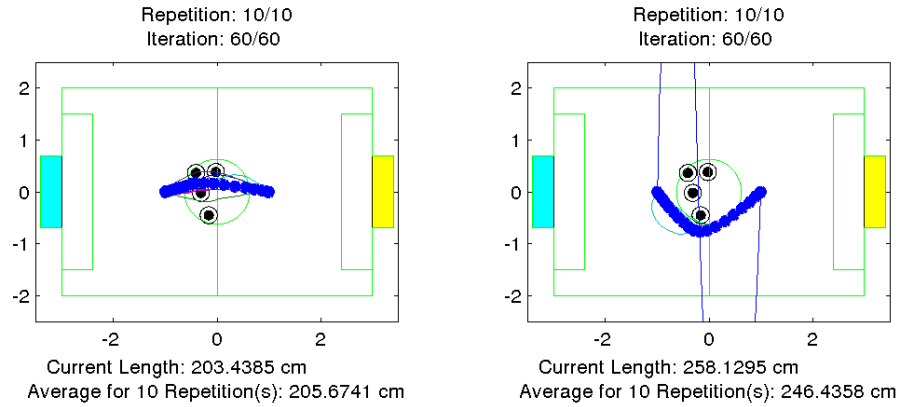

Fig. 3. This is a configuration with 4 obstacles. Although our proposed method's outcome (right side) in the 10th repetition is the safest path, comparing its length with the average length indicates that different repetitions plan different paths (upside or downside of whole obstacles) which means planning safer or the safest path.

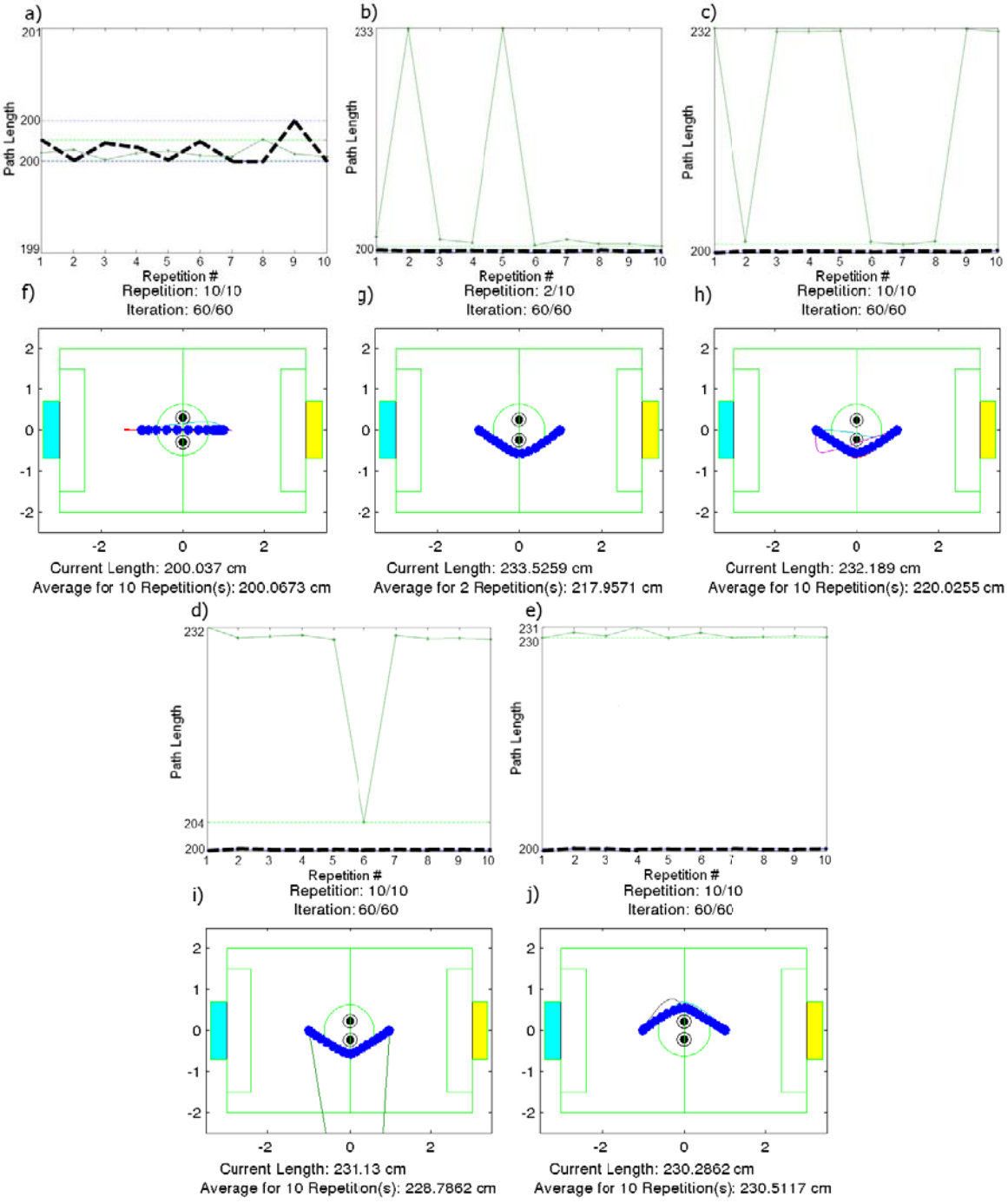

Fig. 4. (a) to (e) illustrate the path lengths versus repetition number in which the thick dashed (mostly over x axis) and thin solid lines correspond to the [8] and our new proposed method, respectively. (f) to (j) are one of their corresponding workspace at a specific repetition.

In the first series of special workspaces we have located two obstacles in initial distance of 60^{cm} (Fig. 4.a and 4.f) then every obstacle is getting closer by1^{cm} in each step. Experimental results shows that the straight line is planned until obstacles are in a 52^{cm} distance to each other. When obstacles are in50^{cm}at 2nd repetitionand 5th repetitions (Fig. 4.b)the planned paths are outside of obstacles which are about 30^{cm} longer. The 2nd repetition of the planned path is presented in (Fig. 4.g). Straight line is planned in more than 80% of repetitions in this distance. When obstacles are in48^{cm}(Fig. 4.c) in 6 out of 10 repetitions, the planned path is 30^{cm}longer. 10th repetition is illustrated in (Fig. 4.h). Herestraight line is planned in about 50% of repetitions. When obstacles are in46^{cm}(Fig. 4.i) straight path is planned in about 10% of repetitions.Fig. 4.d indicates its occurrence just at the 6th repetition. Finally when obstacles

are at a distance of$44^{cm}$and closer (Fig. 4.e and 4.j), the planned path is always a curve outside two obstacles. These results indicate that48^{cm}distance among two obstacles is the point which there is a balance between length and safety. A workspace with farther or closer obstacles leads to straight or curve path planning, respectively. The important point is that the previous method always planes straight path during all steps.

The second series of special workspaces are as previous ones except using three obstacles in two clusters of one and two. The purpose of this space configuration is studying the effects of the number of obstacles. Here the initial distance which indicates the distance of the line connecting center of two near obstacles from another single one should be set to70^{cm}(Fig. 5.a and 5.c) and both of algorithms plan straight paths with length of 200^{cm}. In this case when obstacles are farther than 62^{cm} the planned path is always the straight path. But when obstacles are in 62^{cm}(Fig. 5.b and 5.d) or closer, the straight path through obstacles is never planned; instead in more than 80% of the cases, the path passes near single obstacle and in the remaining 20%, the path is planned near double obstacles. The remarkable point is that the path passing near two-obstacle cluster is about 4% longer. This partial drawback can be solved by increasing the size of swarm which increases the diversity of particles and provides paths with similar fitness. This fact is confirmed by examining this special space configuration with a swarm of 100 particles. In this experiment planned path is near single obstacle from the distance of62^{cm} stably. When obstacles are at 56^{cm}(Fig. 6.a and 6.c) in two repetitions a longer path of 245^{cm} is planned and the remaining planned paths are about236^{cm}. When obstacles are at 44^{cm}(Fig. 6.b and 6.d) the path near single obstacle is approximately as long as the path near double obstacle (with a difference of7^{cm}). Another notable point is the insensitivity of the previous method in [8], to the obstacles closeness during all steps of both of the above special space configurations.

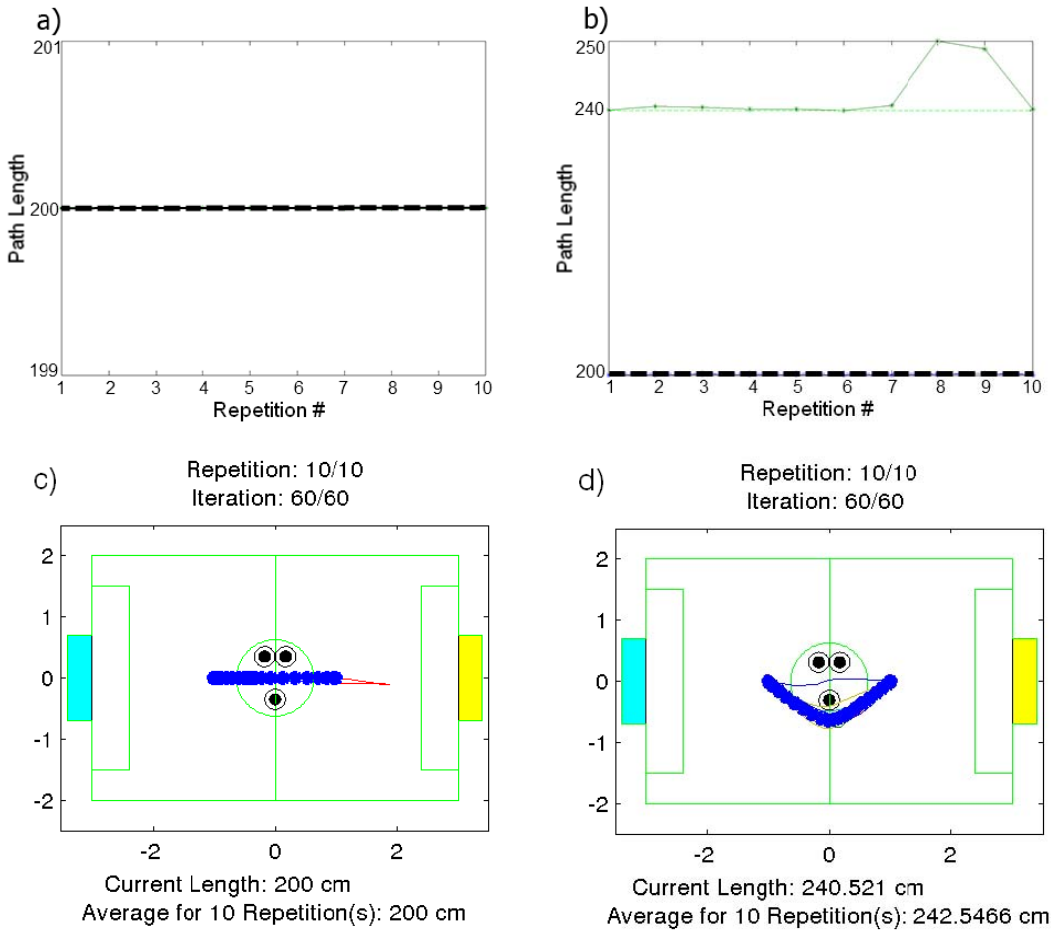

Fig. 5. Second special series of workspaces with 3 close obstacles. (a) and (b) demonstrate path length versus repetition number while figures (c) and (d) display their corresponding space configuration.

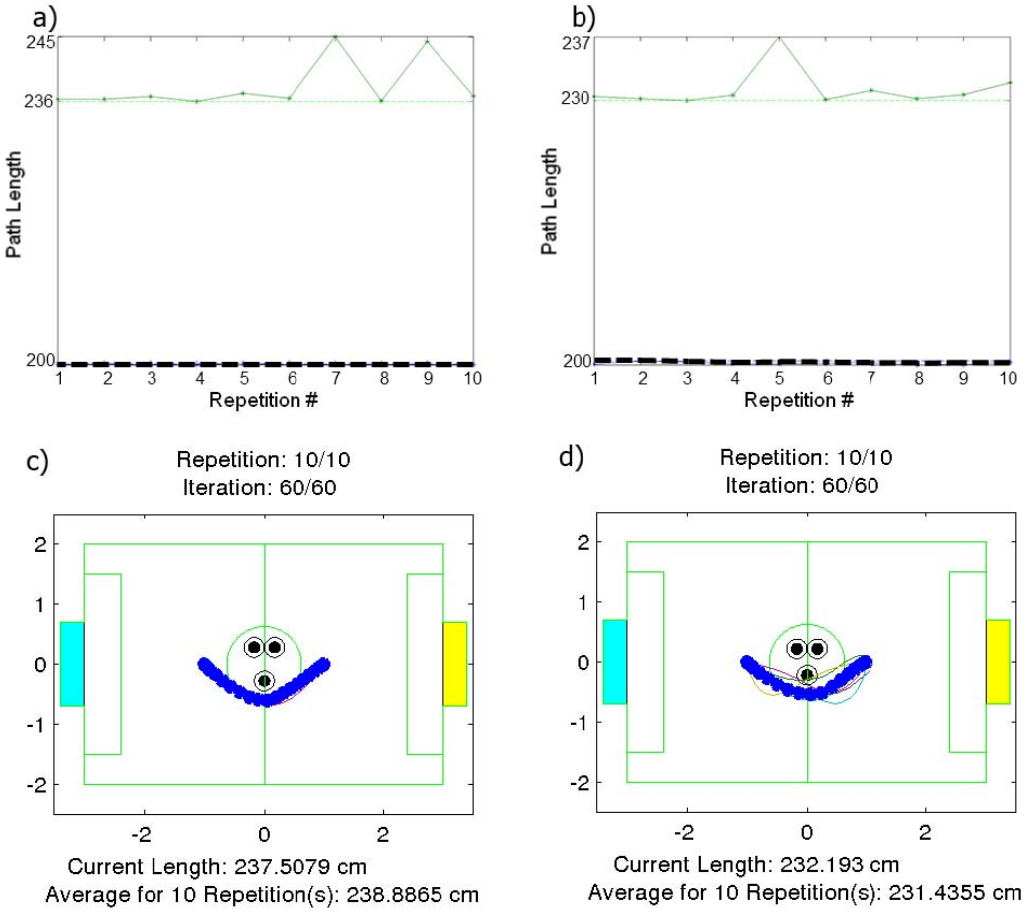

Fig. 6. Continue of second special series of workspaces from Fig. 5. (a) and (b) demonstrate path length versus repetition number and figures (c) and (d) display their corresponding space configuration.

4. Conclusion

In this paper we introduced an improved method for planning a safeand, yet, short path for a humanoid soccer robot. This method uses Ferguson cubic splines for presenting paths and PSO for optimizing paths. Theproposed method plans the shortest and yet safest path at maximum of 60 iterations, with 92% of accuracyand average path length overhead of 14%. Also we investigated obstacles aggregation with the consideration of the balance of safety and length of the path by normalization of fitness function.

References

[1] Liu Yanfei, "Robot path planning based on genetic algorithms with two-layer encoding," Journal of Control Theory and Applications, 2000, vol 17, no. 3, pp. 429-432.

[2] Kennedy J and Eberhart RC, "Particle swarm optimization," Proceedings of the IEEE International Conference on Neural Networks, , 1995, pp. 1942-1948.

[3] Qin Yuanqing, Sun Debao, Li Ning, et al. "Path planning for mobile robot based on particle swarm optimization," Jornal of Robotics and Autonomous Systems, 2004, vol. 26, no.3, pp. 222-225.

[4] Xin Chen and Yangmin Li. "Smooth path planning of a mobile robot using stochastic particle swarm optimization," in Proceedings of the IEEE International Conference on Mechatronics and Automation, 2006, pp.1722-1726.

[5] Saska M, Macas M, Preucil L, et al. "Robot path planning using particle swarm optimization of Ferguson splines," in Proceedings of the IEEE Symposium on Emerging Technologies and Factory Automation. Piscataway, 2006, pp.833-839.

[6] Xianxiang Wu and GuoBaolong, "Mobile robot path planning algorithm based on particle swarm

optimization of cubic splines," Jiriren/Robot, 2009, vol. 31, no. 6, pp. 556-560.

[7] Xianxiang Wu, GuoBaolong, and Lotka-Volterra. "Model based particle swarm optimization, Journal of Control and Decision, 2010, vol. 25, no. 11, pp. 1619-1624.

[8] WU Xianxiang, MING Yan, and WANG Juan, "An improved path planning approach based on particle swarm optimization," International Conference on Hybrid Intelligent Systems (HIS), 2011,pp. 157-161.

[9] Ye J and Qu R., "Fairing of parametric cubic splines," Journal of Mathematical and Computer Modeling, 1999, vol. 30, no. 5/6, pp. 121-131