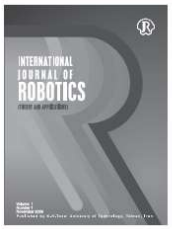


INTERNATIONAL JOURNAL OF ROBOTICS
(THEORY AND APPLICATIONS)
E-mail: IJR@kntu.ac.ir

Robots Positioning Using Particle Swarm Optimization Algorithm

H. Rashidi [a,*]

Department of Mathematics and Computer Science, Allameh Tabataba'i University, Tehran, Iran

ARTICLE INFO

Article history:
Received: 2019-07-20
Received in revised form: 2020-02-07
Accepted: 2020-05-09

Keywords:
Positioning, Inverse Conversion Perspectives, Particle Swarm Optimization, Hough Conversion

ABSTRACT

The positioning of robots, relative to the origin of coordinates, is a crucial issue for autonomous robots. The purpose of the positioning is to find Cartesian coordinates and to position the robot body in a global coordinate system. In this paper, an image-based approach is proposed to robots positioning. In this approach, the pixels for the lines in the ground are specified in the original image. These pixels are replaced by a numerical value of zero and other pixels with a numerical value of one. The image obtained from these points is converted to an image taken from the top view using a reverse perspective transform. Then using the Hough line, the angle of the longest sequence is obtained from the zero values. This angle is used to correct changes resulting from the rotation of the image. In the next step, the information obtained is processed as a matrix containing zero and one using the Particle Swarm Optimization (PSO) algorithm and the coordinates of the location of the robot are determined from the origin. In this paper, an efficient target function is proposed for use in the PSO algorithm. The most important feature of this approach is the use of an Inverse Perspective Map (IPM) transformation to eliminate perspective effects. This method is resistant to changing the size and shape of various forms within the ground. To test the proposed method, positioning for soccer robots is used. The experiments show that the proposed approach has high accuracy in detecting the robot position. The main application of this algorithm is the positioning of airplanes and military missiles based on aerial imagery without the need for a global positioning system (GPS). It can also be used to increase the accuracy of the global positioning system.

1. Introduction

One of the most critical issues in deciding the types of smart robots is to determine their position relative to the origin of coordinates. Determining the precise position of the robot in an area is a very important issue for decisions making. Among all the sensors in the robot, the sight provides the most information in the largest range with very high precision.

This paper is motivated by a need to determine the robot positions using the image captured by the camera located at the head of the human soccer robot. Based on the matching of the lines seen in the image, with the map drawn from these lines in the robot memory. Among the challenges in this area, it can be noted that the shape and size of the rectangles, the circle, and the angle variations of the ground's planes, with respect to the distance from the robot, can be pointed out. The main reason for these changes is the angularity of the robot's camera with the horizon and causing perspective in the image, which makes difficult to detect patterns differently. The other challenge is the wide range of situations that the robot can accommodate so that the

* Corresponding address: Allameh Tabataba'i University, Tehran, Iran
E-mail address: …….Tel: +98212…..

state space is very wide and searching each point in the entire space is impossible in real-time. For example, in a 9-by-6-meter wide pitch, if we want to have a precision of about 5 centimeters, the robot can be located in 21600 different locations, which according to the various angles that can be imagined for the robot. With this regards, 7776000 different image modes can be viewed by the robot in game conditions. This is the case when the angle of the robot with the horizon line is fixed. In this method, firstly, all of the points known as the points of the ground's lines are extracted and the matrix of the position of these pixels is given using the space-time hemogram matrix and converted to an image from the top view. Then the image angle is corrected and the particle swarm optimization algorithm is used to find out the best position that maximizes the target function. The proposed method can be used in various areas, including pattern matching, positioning using images taken by bird defense equipment, and also identifying a specific pattern in satellite imagery.

The remaining sections of this paper are organized as follows. Section 2 reviews the related works. Section 3 presents the proposed method. Section 4 shows the experiments and numerical results. Finally, Section 5 provides the summary and conclusions.

2. Related Works

In this section, we review the latest researches devoted to robot positioning. The probabilistic methods, such as filtering offsets, are the most commonly used methods to solve the problem. The Bayes method uses conditional probability distribution to find the coordinates. Several bay model models can be found to solve this problem, including the Kalman filter([1], [2]), Marco ([1], [3]) and particle filter ([4], [5], [6]). Also, Rossdiner and colleagues presented a positioning method for moving robots based on laser scanner information, which focuses on positioning in two dimensional environments using particle filters. Another approach used to find the position of the robots is a fuzzy approach([8], [9]). In this approach, the robot has, at any time, a belief in its location, which is not believed to be a single position, but rather as a fuzzy set defined in the state of possible situations. Each position has a degree of membership, which indicates its degree of close proximity to the actual position. One of the most popular methods is to find the Monte Carlo method (MCL). This method has the ability to solve local and global positioning problems. The Monte Carlo method uses particle filtering for positioning ([10], [11]). The particle filter is believed to be limited by a limited number of particles. Each particle is a hypothesis about being in a particular position of the space of all positions. The accuracy and computational load of the algorithm depend on the number of particles. Usually increasing the number of particles, the accuracy increases, and computational efficiency decreases. In this model, each particle has a weight that indicates the magnitude of that particle, and the sum of all these weights is equal to one.

Klus et al discussed the issues of determining the error of industrial robots positioning repeatability [12]. A neural mathematical model that allows for predicting its value with the error of less than 5% was designed. The obtained results were compared to a classical mathematical model. It was revealed that a well-trained neural network enables the prediction of the error of positioning repeatability with the doubled accuracy.

Utama et al. introduced the implementation and the method to control the robot soccer which uses omnidirectional wheels [13]. Forward, inverse kinematics algorithms and PID controller have also used an arm controller, encoder and compass sensors are utilized. This paper shows that the proposed system has the potential to control the position and maneuver of the robot.

Fernandes et al. presented a comparison of four nonlinear controllers for positioning tasks of a unicycle-like mobile robot [14]. The controller performance is analyzed through three indexes: IAE, ITAE and IASC. The analysis is performed experimentally. Initially, the controller gains are tuned by simulation until IAE index is less than 5 percent comparatively. Next, the controllers are run in practice and ITAE and IASC indexes are analyzed. The objectives are to analyze the convergence time to the desired goal and the energy consumption for the mission accomplishment. The controller stability is demonstrated in the sense of Lyapunov and validated experimentally.

Li et al. discussed the method of positioning the ball in soccer robot competition [15]. By analyzing the movement of the ball in relation to the robot goalkeeper, which could be divided into two major types, a positioning method for robot soccer based on Kalman filter is proposed. In the paper, the feasibility of this method is analyzed and its effectiveness is proved by experiments. The positioning method based on Kalman filter could make it more accurate to determine the location of the ball and it is beneficial for the robots to make correct decisions.

Chaofeng et al. used ant colony algorithm and ultrasonic positioning method to research the driving path of the car, for planning effectively driving path, positioning precisely and reducing position error in the driving process [16]. In the paper, the departure and destination of the car are set and the driving path change trend of the car is analyzed with different numbers of ants and numbers of iterations, according to the traffic information instructed by the map, which is input to the master chip of the car. Using the method of

ultrasonic positioning, the reaction speed in the driving process of the car is studied and the effect of ultrasonic positioning method to the time of repositioning is discussed. The research shows that the optimal positioning time is 8.56s when the number of ants is 30 and the number of traversal is 50. Combined with ultrasonic positioning method, it can accurately determine whether the car reach the destination in accordance with the established goals. Through the research of this paper, it is accurate to get the driving position of the car.

3. Proposed Method

The proposed method of robot positioning is depicted in Fig. 1. This method consists of four main steps as follows: (a) Take the input image and extract points related to the ground lines; (b) Apply the inverse perspective map (IPM) transformation; (c) Determine the rotation angle of the robot and (d) Apply the algorithm PSO.

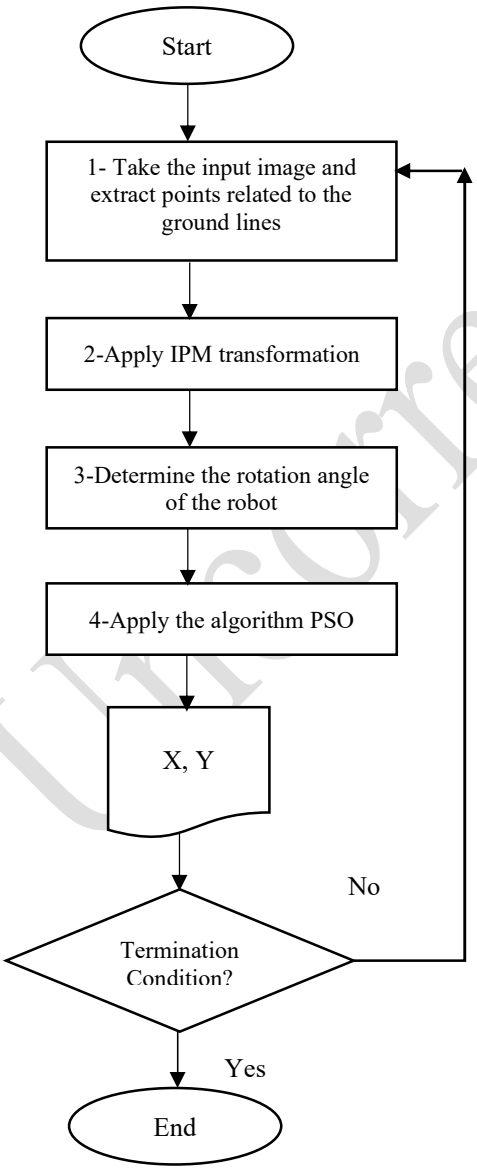


Figure 1: The flowchart of the proposed method

3.1. Take the Input Image and Extract Points Related to the ground lines

The first step to reducing the processing load on the conversion agents IPM as well as noise reduction, where white on a green background with landlines are considered and the coordinates of the points are stored. The stored points are only places that used for the next step in the IPM transformation. Figure 2 shows the image after processing.

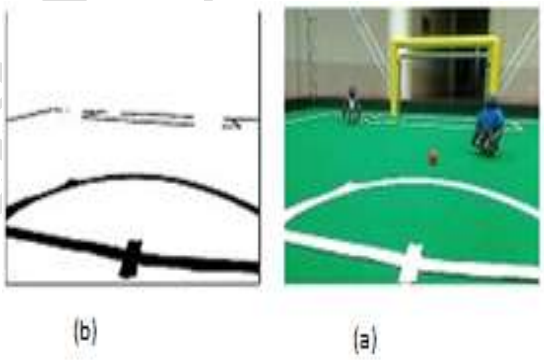


Figure 2: (a) Image Taken by the camera is located at the head of the robot (b) Image Obtained Later after removing the additional points

3.2. Apply IPM Transformation

The raw image taken by the camera on the robot head is not suitable for positioning because of the presence of perspective phenomena. It is, therefore, highly desirable that the image captured by the image be converted from the top view. The main feature of this image is the constant size of the shapes and dimensions of the lines in which it exists. For this reason, a homograph matrix is used that shows the relation between the two points in space with $m' = Hm$, where m and m' are determined by Equation (1).

$$m = (x, y, 1), m' = (x', y', 1) \qquad (1)$$

The matrix H is also a 3 × 3 matrix called the Homograph Matrix, as Equation (2).

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ 1 \end{bmatrix} \sim \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{22} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad (2)$$

Using this matrix, each point can be moved and rotated in space, relative to the other points. If the coordinates of all the pixels of an image are converted by this matrix, one can create an image of another, converting it into a bird's eye view or from the top view. Figure 3 shows the actual camera location and the virtual camera after the conversion.

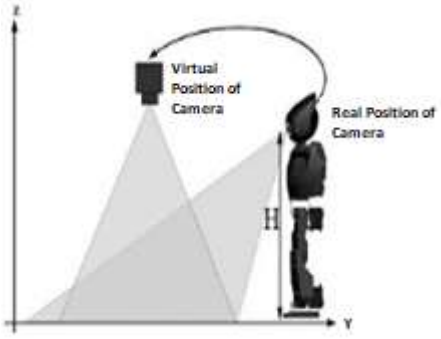


Figure 3: Change the angle of the camera using a homogeneous matrix

Equations (3), (4), and (5) have been used to obtain the new position of pixels.

$$H_{x^*} = H\frac{x\sin\theta + f\cos\theta}{-y\cos\theta + f\sin\theta} + d \quad (3)$$

$$H_{y^*} = H\frac{y\sin\theta + f\cos\theta}{-y\cos\theta + f\sin\theta} + d \quad (4)$$

$$Hd = \left|\frac{H(\sin\theta + f\cos\theta)}{f\sin\theta - \cos\theta}\right| + 1 \quad (5)$$

Here, H is the height of the main camera from the ground, y, x, the position of a pixel in the image and x*, y* are the same pixel position after the transfer. θ is the angle of the camera with the horizontal line and f is the focal length of the camera. In Figure 4, the conversion result is shown on the points in the line.

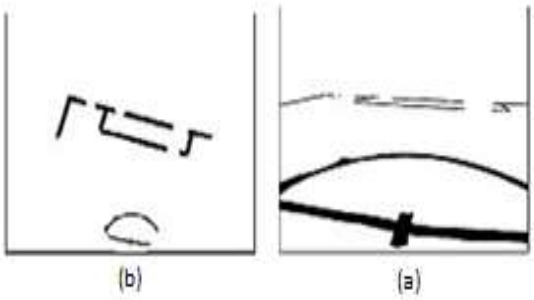


Figure 4: (a) The image obtained after the removal of pixels that are not related to the ground lines. (b) Image after applying IPM transformation.

3.3. Determine The Rotation Angle of the Robot

Given that the robot can be placed at different angles at any one time, the resulting image can be found depending on the rotational rotation of the robotic era, which is the time of the image, which extends over the state of the space. To remove this period from the image, the image must be reversed in the opposite direction of the robotic era. In this case, the rotation angle of the robot is required. The easiest way to use compass sensors in a robot is to determine the angle of the robot with respect to the magnetic properties of the ground. Of course, the basic problem with these sensors is their high sensitivity to noise, which reduces accuracy by about 15-15 degrees. Another method is to use the angle of the lines shown in the image. To obtain this angle, the Hough line algorithm is used to determine the angle of all the lines in the image, along with the arrangement of these lines from the largest to the smallest. Of course, in this algorithm, only the tallest line angle of at least 30 pixels is used. The angle of these lines is very precise after the IPM stage, but the main problem is that the robot cannot determine which linear angle it is detected is related to the horizontal line of the ground, either vertically or even none of them. Therefore, to solve this problem, the angles taken from the compass sensor and the angle obtained from the image are simultaneously used. The difference between these two angles is calculated. Since there are only horizontal or vertical lines in the image, we must determine that the obtained angle is related to those horizontal or vertical lines. Then the proportional image is rotated. When using military sensors, it can be used directly from the angle of the sensors.

3.4. Apply the algorithm PSO

The Paticle Swarm Optimization was coined by Kennedy and Eberhart in 1995, based on swarm congestion in the natural environment [16] . Figure 5 shows the pseudo-code of this algorithm.

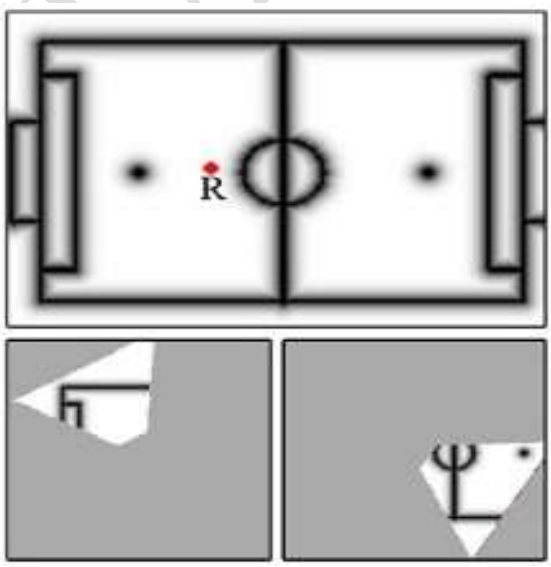
```
[x*] = PSO();
P = Particle_Initialization(); // Initialize swarm particles
For i=1 to it_max
  For each particle p in P do
    fp = f(p);
    If fp is better than f(pBest)
        pBest = p;
    end
```

Figure 5: The pseudo-code of PSO.

The variables/parameters used in Figure 5, along with their description, are as follows:

- *p*: particle's position
- *it_max*: The maximum number of iterations
- *f*: The Objective function
- *v*: path direction
- c_1: the weight of local information
- c_2: the weight of global information
- *pBest*: the best position of the particle
- *gBest*: the best position of the swarm
- *rand*: random variable

PSO is one of the optimization methods of nature-inspired optimization. This algorithm uses random numbers and general relations between particles. Because it does not need to encode and code parameters to binary strings, as it is in the genetic algorithm, its implementation is crucial. This algorithm searches for the space of a target function by setting the particle path (as separate agents), moving a particle in an area that moves collectively with other particles. It has two main components (a random component and one deterministic component). Each particle moves towards the best overall position ever experienced by the entire population and the best place it has ever had, while at the same time it does not lose its willingness to perform a random motion. The goal is to find out the best current position among all the particles, until an improvement is achieved or the certain number of repetitions are performed.



Considering that the robot field of vision is 120 degrees and is placed at angles anywhere in the field, even after converting the IPM and smoothing the image, even with the minimum accuracy required for this problem (the robot can accurately measure 5 cm of its location The total number of different images in a 9 to 6-meter plot is 7,776,000, according to the 360 degrees angle variation. If we only need 100 to 100 pixels per image, we must compare nearly 80 billion views every time. Even with this comparison, we cannot be sure that the correct solution has been found. In Figure 6, several images that the robot can observe in one place is shown.

Figure 6: Two examples of images that the robot can only see by changing the angle.

As can be seen in Fig. 5, the robot R can observe different images at a fixed point. This field of view makes it difficult to locate the robot, so there is a perfectly intelligent way to overcome this challenge in solving this problem. To solve the problem, the PSO algorithm is used.

In the PSO algorithm, many particles randomly distributed over the whole space are used, where new particle positions are obtained based on Equations (6) and (7).

$$v[t+1] = \omega * v[t] + c_1 * r_1 * (pb[t] - x[t]) + c_2 * r_2 * (gb[t] - x[t]) \quad (6)$$

$$x[t+1] = x[t] + v[t+1] \quad (7)$$

In the equations, ω is the coefficient of inertia, r_1 and r_2 are random numbers with uniform distribution and in intervals zero to one, as well as c_1 and c_2 learning coefficients. The r_1 and r_2 make a variation in the solution, thus a complete search in space are made. The c_1 is the coefficient of learning for personal experiences and the c_2 is the coefficient of learning associated with collective experiences. The *pb* is the best success experienced by each particle and the *gb* is the best-experienced experience of the whole particle. As a result, with respect to these relationships, each particle takes its movement based on three parameters: (a) the previous direction of movement; (b) the best position in which it is located; (c) the best position experienced so far. But to use the PSO, we need to have an objective function that can be used to measure the utility at any given time. A two-dimensional array has been used to compute the target function, known as the matrix w, which contains values between -1 and +1. The dimensions of this array are proportional to the ground

of the tournament. This two-dimensional array is formed with the desired accuracy of 5 centimeters and real dimensions of 9×6 meters, with an additional 70 centimeters of margin around the ground, which contains 208 × 148 points. In the places where the line is located, the amount of the corresponding matrix corresponding to that point 1, and the rest of the layers, taking into account the distance that those points from the nearest point of a line on the ground, take between -1 and +1, which is obtained from equation (8) comes.

$$p = \begin{cases} \left(\frac{\max(d) - d}{\max(d)}\right)^4 & d \le 10 \\ -1 & d > 10 \end{cases} \quad (8)$$

The distance of a point in the array with the closest point in the line and the max (d) is equal to the maximum tolerable distance, which is considered here 10. Figure 7 shows the matrix w after weighing. At any moment, the robot can see its 4.5-meter distance accurately.

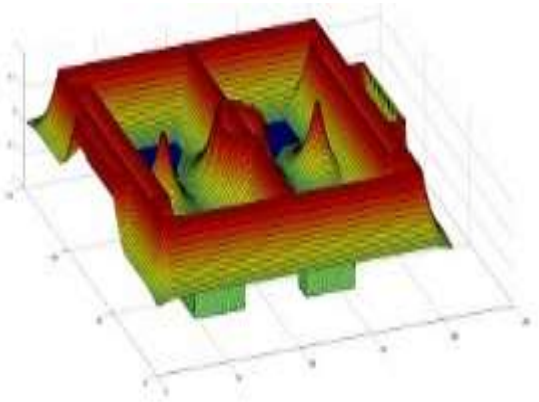


Figure 7: Matrix w after weighting

At any moment, regardless of the angle of the robot, one can consider a circle of 4.5-meters radius as a robot's robustness range. This circle is considered in 180 × 180 dimensional arrays (radius of 4.5 meters, which creates a circle of 9 meters in diameter, thus, taking into account the accuracy of 5 cm, this circle can be considered in a square to the side of 180 members). The points that are detected as lines in the image are set to one in this array and the other points are considered zero. We call this matrix the search matrix and represent it as p. The steps for creating the search matrix are shown in Figure 8. The search matrix p has a scale such as the matrix w and the value of the function for the fit function at the point x and y of the matrix w is calculated using the equations (9), (10), (11).

$$n(x,y) = \sum_{i=-90}^{90} \sum_{j=-90}^{90} p(i,j)\omega(x+i,y+j) \quad (9)$$

$$ns = \sum_{i=-90}^{90} \sum_{j=-90}^{90} p(i,j) \quad (10)$$

$$f(x,y) = \frac{n(x,y)}{ns} \quad (11)$$

The equation (11) has a value between -1 and +1, and indicates the degree of matrix matching p and matrix w. Because the points near the lines are also set in the matrix w in accordance with their location, even if the matrix p has small displacements or angle variations of 6 degrees. In this case, the value of the function f in the point on the map where the robot is located is maximized.

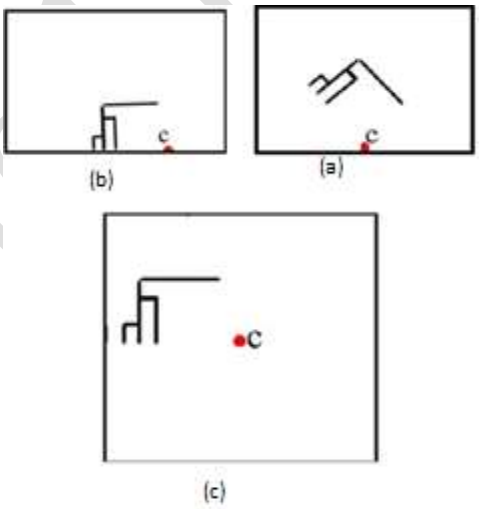


Figure 8: The steps to create a search matrix

Figure 9 shows the different values of Equation (11) for an image for the placement of the robot at all points where possible, in fact, this equation has calculated the correlation of the input image with the map of the ground in different locations. Also, with respect to the negative values in the matrix w, if the values of one in the matrix p, are set to negative values in the matrix w, the output value of the equation (9) is reduced. In fact, the main reason for the existence of the range of variations from -1 to +1 in the matrix w is that the output values in Equation (9) fall in the wrong areas and increase in areas close to the solution.

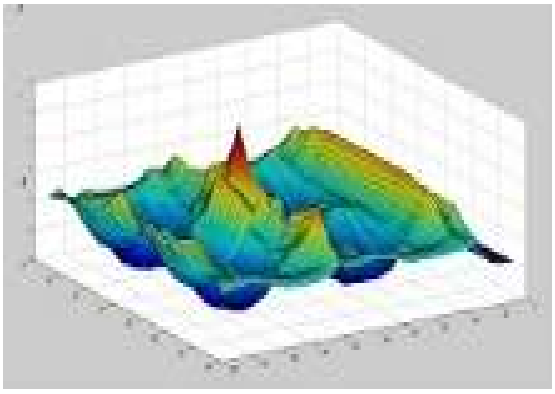


Figure 9: The values of the f function for all possible points where the robot can fit.

4. Experimental Results

We implemented the algorithm in MATLAB platform. Although the typical number of particles for PSO is between 20 and 40 ([18], [19]), our experiment by trial and error showed that 50 particles randomly distributed over the whole space make a better solution. The results of implementing the four steps of the proposed method are shown in Fig. 10, respectively. This method usually finds one of the optimal points in less than 40 repetitions, which, in 90 percent of the time, takes into account the values of c_2 and c_1 equal to 2, and ω = 0.8 in this case, it succeeds in finding a global optimum. Figure 10 shows the search results in the four cases of the repeating of this algorithm.

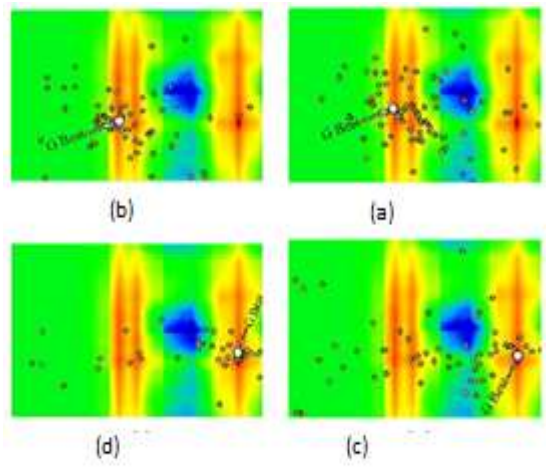


Figure 10: (a) PSO in the first repeat, (b) after 10 repeats, (c) after 20 repeats and (d) after 40 repeats.

In order to test this algorithm, football robots were used as one of the applications of this method in the center research of Qazvin Islamic Azad University. To this end, the robot was started for 94 minutes in a simulation environment with 12.5 frames per second. The dimensions of the playground were 9×6 meters, with an additional 70 centimeters of margin around the ground, which contains 208 × 148 points. A total of 56558 image frames (100 to 100 pixels per image) were evaluated for each real-frame location and the angle of the robot was obtained through the simulation. These values were stored along with the values extracted by the proposed method. We executed the software for 20 rounds. In each round, the position errors and angle errors are recorded. The Fig. 11 and Fig 12 show the average errors of the position and angle, respectively, in each round.

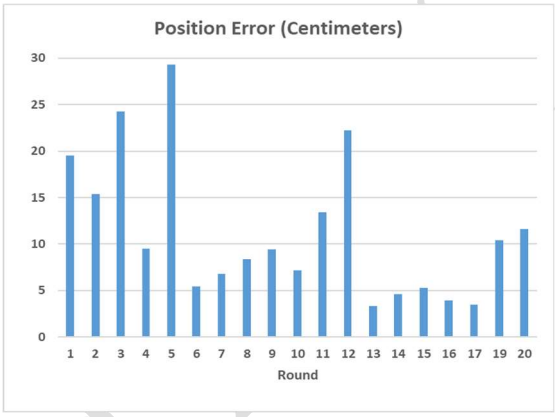


Figure 11: The average position error for 20 rounds of running the software.

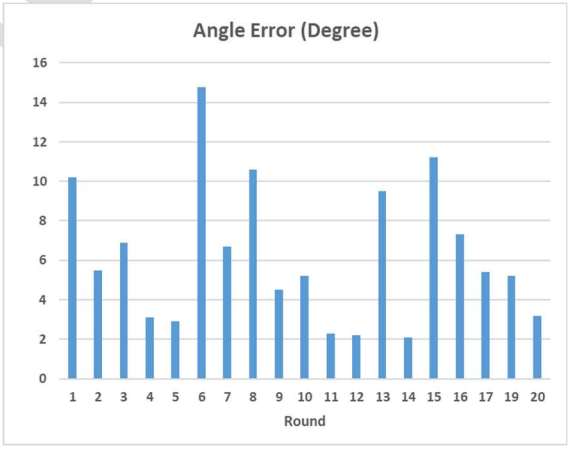


Figure 11: The average angle error for 20 rounds of running the software.

Table 1 shows the means of the errors. In the aggregation, in 52827 (%94) times the robot successfully finds the correct positioning and could not find its correct position in 3731 times (%6) (an error of more than 30 cm and an angle of more than 15 ° was considered as an inability to be positioned). Table 1 shows the results of the algorithm evaluation at 56558 frames.

Table 1: Estimated position error

Error	Mean	Maximum
Position error (centimeters)	11.23	29.35
Angle error (degree)	6.25	14.78

Our experiment shows that with using the particle swarm optimization algorithm, the number of calculations for this function can be reduced for each positioning. Table 2 shows a comparison of two approaches using the particle swarm optimization algorithm and without using this algorithm.

Table 2: A comparison of the number of calculations of the fit function

Approach	The number of calculations of the function f - Equation (11)
Without using PSO	21600
With using PSO	2000 (max)

The first approach is 'without using PSO' in which we don't use any optimization algorithm. In fact, in the approach, we use the greedy search algorithm that needs many comparisons to find the best solution.

5. Summary and Conclusion

In this paper, an effective approach based on particle swarm optimization algorithm was developed. In this approach, the inverse of the perspective is used to increase the speed and reduce the size of input information. The dimensions of the matrix w shown in Fig. 6, can be determined by the environment in which the positioning is performed. This matrix is weighted in the light of the edge-finding information in other applications.

In this paper, the main objective was to find out the best position with a precision of 5 cm for the soccer robots. If the precision is reduced to one meter, the algorithm can process much larger matrices in real-time. The equation (11) is also used to find the position of a robot, which is calculated in non-smart methods of this function for the number of places where a robot may be located. Using the particle swarm optimization algorithm, we can reduce the calculation of this function for each positioning. This method has high accuracy due to the use of points in the ground lines for positioning and can also detect the robot position with some of these points, which is therefore highly resistant to noise. One of the important applications of the proposed approach is the positioning of minesweeper robots, planes and missiles with regard to aerial imagery and without the need for a global positioning system (GPS). If the signals sent via the GPS are encountered an error or deliberate error, this method will provide a good security for military and civilian navigation systems to determine the location.

References

[1]. Leonard, J.J. and Durrant-Whyte, H.F, Mobile robot localization by tracking geometric beacons, in Robotics and Automation, IEEE Transactions on, (1991), pp. 376–382.

[2]. Vitali, L. Jetto and S. Longhi and D., Localization of a wheeled mobile robot by errorsensor data fusion based on a fuzzy logic adapted Kalman filter, in Control Engineering Practice, (1999), pp. 763-771.

[3]. Thrun, Dieter Fox and Wolfram Burgard and Sebastian, Active Markov Localization for Mobile Robots, in Robotics and Autonomous Systems, (1998), pp. 195-207.

[4]. Thrun, Dieter Fox and Wolfram Burgard and Frank Dellaert and Sebastian, Monte Carlo Localization: Efficient Position Estimation for Mobile Robots, (1999), pp. 1-7.

[5]. Dellaert, Sebastian Thrun and Dieter Fox and Wolfram Burgard and Frank, Robust Monte Carlo localization for mobile robots, in Artificial Intelligence, (2001), pp. 99-141.

[6]. Valera, C. Sánchez and A. Soriano and M. Vallés and E. Vendrell and A., Geometrical matching analytical algorithm for fast mobile robot's global self-localization, in Robotics and Autonomous Systems, Vol. 62 (6), (2014), pp. 855-863.

[7]. Rusdinar, A. and Jungmin Kim and Sungshin Kim, Error pose correction of mobile robot for SLAM problem using laser range finder based on particle filter, in Control Automation and Systems (ICCAS), 2010 International Conference on, (2010), pp. 52 – 55.

[8]. Saffiotti, D. Herrero-Pérez and H. Martínez-Barberá and K. LeBlanc and A., Fuzzy uncertainty modeling for grid based localization of mobile robots, in International Journal of Approximate Reasoning, Vol. 51(8), (2010), pp. 912-932.

[9]. Buschka, P. and Saffiotti, A. and Wasik, Z, Fuzzy landmark-based localization for a legged robot, in Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference, (2000), pp. 1205-1210.

[10]. Wang, Xiaohuan Lu and Yuning Dong and Xinheng, A Monte Carlo Localization algorithm for 2-D indoor self-localization based on magnetic field, in Communications and

Networking in China (CHINACOM), 8th International ICST Conference on, (2013), pp. 563-568.

[11]. Fox, Dieter, Adapting the Sample Size in Particle Filters Through KLD-Sampling, in International Journal of Robotics Research, (2003), pp. 1-27.

[12]. Kluz, R., Antosz, K., Trzepieciński, T., Gola, A., Predicting the error of a Robot's positioning repeatability with artificial neural networks, Advances in Intelligent Systems and Computing, Vol. 1004, (2020), pp. 41-48.

[13]. Utama, K.V., Fatekha, R.A., Prayoga, S., Pamungkas, D.S., Hudhajanto, R.P., Positioning and Maneuver of an Omnidirectional Robot Soccer, Proceedings of the 2018 International Conference on Applied Engineering, ICAE 2018, (2018), 8579148.

[14]. Fernandes, R., Bessa, M.M., Brandao, A.S., Performance analysis of positioning controller for mobile robots, Proceedings - 2017 LARS 14th Latin American Robotics Symposium and 2017 5th SBR Brazilian Symposium on Robotics, LARS-SBR 2017 - Part of the Robotics Conference 2017, (2017), pp. 1-5.

[15]. Li, J., Xiao, J., Xu, S., Positioning method for robot soccer based on Kalman filter, IEEE International Conference on Control and Automation, ICCA, 8003179, (2017), pp. 892-897.

[16]. Chaofeng, L., Xuemei, S., Shimei, Y., Shouqiang, K., Chuang, H., The research of robot toy car positioning method based on ant colony algorithm and ultrasonic positioning, 1st International Conference on Electronics Instrumentation and Information Systems, EIIS 2017, (2018), pp. 1-4

[17]. Kennedy J., Eberhart R., Particle Swarm Optimization, in Proceeding IEEE International Conference on Neural Network, Vol. 4, (1995), pp 1942-1948.

[18]. Krastev T., Particle Swarm Optimization (PSO) open source framework, available at https://bee22.com/, Last checked: 6 Feb 2020.

[19]. Turkedjiev E., Hybrid neural network analysis of short-term financial shares trading. Doctoral Thesis (2017), Northumbria University.

Biogaphy:

Hassan Rashidi is a Professor in Department of Mathematics and Computer Science of Allameh Tabataba'i University. He received the B.Sc. degree in Computer Engineering and M.Sc. degree in Systems Engineering and Planning, both from the Isfahan University of Technology, Iran. He obtained Ph.D. from Computer Science and Electronic System Engineering department of University of Essex, UK. His research interests include software engineering, software testing, and scheduling algorithms. He has published many research papers in International conferences and Journals.