Numerical evaluation of mobile robot navigation in static indoor environment via EGAOR Iteration

A A Dahalan\textsuperscript{1}, A Saudi\textsuperscript{2}, J Sulaiman\textsuperscript{3}, W R W Din\textsuperscript{1}

\textsuperscript{1} Department of Mathematics, Centre for Defence Foundation Studies, National Defence University of Malaysia, 57000 Kuala Lumpur, Malaysia
\textsuperscript{2} Faculty of Computing and Informatics, Universiti Malaysia Sabah, 88400 Kota Kinabalu, Sabah, Malaysia
\textsuperscript{3} Faculty of Science and Natural Resources, Universiti Malaysia Sabah, 88400 Kota Kinabalu, Sabah, Malaysia

E-mail: a.qilah@upnm.edu.my

Abstract. One of the key issues in mobile robot navigation is the ability for the robot to move from an arbitrary start location to a specified goal location without colliding with any obstacles while traveling, also known as mobile robot path planning problem. In this paper, however, we examined the performance of a robust searching algorithm that relies on the use of harmonic potentials of the environment to generate smooth and safe path for mobile robot navigation in a static known indoor environment. The harmonic potentials will be discretized by using Laplacian’s operator to form a system of algebraic approximation equations. This algebraic linear system will be computed via 4-Point Explicit Group Accelerated Over-Relaxation (4-EGAOR) iterative method for rapid computation. The performance of the proposed algorithm will then be compared and analyzed against the existing algorithms in terms of number of iterations and execution time. The result shows that the proposed algorithm performed better than the existing methods.

1. Introduction

In recent years, the applications of path planning such as the movement of robots and autonomous agents has become a popular research topic. Generally, mobile robot path planning is about seeking a collision-free motion through the described environment with obstacles in order to reach a particular position. In this paper, based on the theory of heat transfer, the mobile robot path planning will be implemented via numerical potential function in configuration space. By using Laplace’s equation, the heat transfer problem is modelled and the solutions of the equation are the harmonic functions. For simulation of paths derived from harmonic functions, the temperature values in the configuration space will be used. Due to the availability of fast processing machine and efficiency in solving the problem, numerical techniques is been used to obtain harmonic functions. This paper conducted several experiments to study the performance of 4-Point Explicit Group Accelerated Over-Relaxation (4-EGAOR) iterative method for generating mobile robot path in several sizes of environment with varying number of obstacles.

2. Related Work

The use of potential functions for robot path planning was introduced by Khatib [1]. It views every obstacle to be exerting a repelling force on an end effector, while the goal exerts an attractive force. Meanwhile, Connolly et al. [2] and Akishita et al. [3] independently developed a global method using Laplace’s equations for path planning to generate a smooth collision-free path. Both works shows that
harmonic functions offer a fast method of producing paths in a robot configuration space and prevents the spontaneous creation of local minima. Then, Sasaki [4] demonstrated the use of numerical technique for solving path planning problem. It claims that the new computational approach to motion planning worked quite successfully through simulation of complicated maze problems. Karova et al. [5] presented Dijkstra’s algorithm using image processing for mobile robot path planning in a labyrinth. The algorithm finds the shortest path to a final target and shows it’s able to move an object in a labyrinth with large size for a minimum number of time. While Hachour [6] proposed navigation of an autonomous mobile strategy designed in a grid-map form of an unknown environment with static unknown obstacles using hybrid intelligent. The main feature of it is the use of the best path of biological genetic principle combined with networks in the task fuzzy reasoning and inference capturing human expert knowledge to decide about the best avoidance direction getting a big safety of obstacle danger. The works has demonstrated the simulation by using two programming languages: in visual basic language the robot reaches the target by avoiding all of the obstacles whilst in Delphi language, the robot takes the shortest path to reach the target.

In the previous studies, the standard numerical technique such as Gauss-Seidel (GS) and Successive Over-Relaxation (SOR) iterative methods were tested in computing the harmonic potentials [2,4,7,8]. Recently, block variants of SOR method were employed for faster computation [9,10]. In addition to motion planning of autonomous mobile robot, harmonic potentials were also applied to many other applications such as UAV motion planning [11,12], marine vessel path planning [13], ship navigation [14], trajectory control [15], space exploration [16], etc.

3. Methodology

In the study of heat conduction, the Laplace’s equation also known as the steady-state heat equation [17]. Mobile robot path planning problem can be modelled as a heat transfer problem and the solutions of Laplace’s equation are the harmonic functions. In the configuration space containing obstacles, the harmonic potentials is computed over the entire region and the harmonic solutions are used to find a path lines from initial point to the goal point for a mobile robot. The goal point is treated as a sink pulling heat in. Whereas the outer boundaries, inner walls and obstacles are considered as heat sources that are fixed with constant temperature values. As the result of a heat conduction process, a temperature distributions, that represent the Laplacian potential values, develops and the heat flux lines that are flowing to the sink, fill the configuration space. The path can then be easily found by following the heat flux. This process guarantees a path to the goal without encountering local minima and successfully avoiding any obstacles as observed by Connolly et al. [2]

Mathematically, a harmonic function on a domain \( \Omega \subset \mathbb{R}^n \) is a function which satisfies Laplace’s equation, in which \( x_i \) is the \( i \)-th Cartesian coordinate and \( n \) is the dimension. In the case of robot path construction, the domain \( \Omega \) consists of the outer boundaries, obstacles, start points and goal point.

\[
\nabla^2 \phi = \sum_{i=1}^{n} \frac{\partial^2 \phi}{\partial x_i^2} = 0
\]

The Laplacian, equation (1) can be solved efficiently via numerical method. Various block iteration techniques were implemented producing impressive performance [18,19]. In this model, the robot is represented by a point in the configuration space. The configuration space is designed in grid form and the coordinates and function values associated with each node are computed iteratively by applying numerical technique to satisfy equation (1). The start point is assigned with high fixed potential value whereas the goal point is the lowest fixed assigned potential value, meanwhile different initial temperature values are assigned to the outer wall boundaries and obstacles. In this work, solution to the Laplace’s equation were subjected to Dirichlet boundary conditions, \( \Phi | \partial \Omega = c \), where \( c \) is constant. Once the harmonic function under the boundary conditions is established, the required path can be easily found by following the heat flux performed by gradient descent strategy on computed potential values. This descending search to a succession of points with lower potential values leading to the point with the least potential value that represents the goal point. The coordinates and the nodal gradients of temperature obtained from the finite difference analysis can be used to draw the path.
4. Formulation of 4-Point Explicit Group Accelerated Over Relaxation (4-EGAOR) Iterative Method

In the robotics literature, the standard GS [2] and SOR [8-10] have been used to solve equation (1). To compute the solutions of Laplace’s equation (1), this study proposed faster numerical solver by employing 4-EGAOR iterative method. Previous works [18-21] on block iterative methods utilize various points of Explicit Group (EG) methods and they proved that block iterative method is more superior compared to the traditional point iterative methods.

Consider the two-dimensional Laplace’s equation in equation (1) defined as
\[ \nabla^2 U = \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} = 0. \]  

(2)

Equation (2) can be simplify to the five point second-order standard finite difference approximation equations as generally stated in the following equation which also known as standard Gauss-Seidel iterative method for solving linear system

\[ u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j} = 0. \]  

(3)

To enhance convergence speed, an approach called Acceleration Over-Relaxation (AOR) method is added to equation (3) as shown below

\[ u_{i,j}^{(k+1)} = \frac{r}{4} (u_{i-1,j}^{(k)} - u_{i,j-1}^{(k)} + u_{i+1,j}^{(k)} - u_{i-1,j}^{(k)}) + \frac{\omega}{4} (u_{i-1,j}^{(k)} + u_{i+1,j}^{(k)} + u_{i,j-1}^{(k)} + u_{i,j+1}^{(k)}) + (1 - \omega) u_{i,j}^{(k)} \]  

(4)

where \( r \) and \( \omega \) are the optimum relaxation parameters and \( \omega \) is defined in the range of \( 1 \leq \omega < 2 \). For AOR method, there is no general formula in getting the minimum number of iterations by determining the optimum values of \( r \) and \( \omega \). According to Hadjidimos [22] the value \( r \) is normally chosen to be close to the value \( \omega \) of the corresponding SOR.

To examines the effectiveness of the block iterative method, consider a four uniform solid nodal points and defined as

\[
\begin{pmatrix}
4 & -1 & -1 & 0 \\
-1 & 4 & 0 & -1 \\
-1 & 0 & 4 & -1 \\
0 & -1 & -1 & 4
\end{pmatrix}
\begin{pmatrix}
u_{i,j} \\
u_{i+1,j} \\
u_{i,j+1} \\
u_{i+1,j+1}
\end{pmatrix}
= \begin{pmatrix}S_1 \\
S_2 \\
S_3 \\
S_4\end{pmatrix}
\]  

(5)

where

\[ S_1 = u_{i-1,j} + u_{i,j-1}, \]
\[ S_2 = u_{i+1,j} + u_{i+1,j-1}, \]
\[ S_3 = u_{i,j+1} + u_{i+2,j}, \]
\[ S_4 = u_{i+1,j+1} + u_{i+1,j+2}. \]

Determining the inverse matrix of the coefficient matrix in equation (5), the general scheme of the block iterative method can be rewritten as [19]

\[
\begin{pmatrix}
u_{i,j} \\
u_{i+1,j} \\
u_{i,j+1} \\
u_{i+1,j+1}
\end{pmatrix}
= \frac{1}{24}
\begin{pmatrix}
7 & 2 & 2 & 1 \\
2 & 7 & 1 & 2 \\
2 & 1 & 7 & 2 \\
1 & 2 & 2 & 7
\end{pmatrix}
\begin{pmatrix}S_1 \\
S_2 \\
S_3 \\
S_4\end{pmatrix}
\]  

(6)

The implementation of equation (6) can be processed iteratively to compute the value of four node points simultaneously, as shown below
\[ u_{i,j}^{(k+1)} = \frac{1}{24} \left( 7S_1 + 2S_2 + 2S_3 + S_4 \right) + u_{i,j}^{(k)}, \]
\[ u_{r_{i,j}}^{(k+1)} = \frac{1}{24} \left( 2S_1 + 7S_2 + S_3 + 2S_4 \right) + u_{r_{i,j}}^{(k)}, \]
\[ u_{i,j+1}^{(k+1)} = \frac{1}{24} \left( 2S_1 + S_2 + 7S_3 + 2S_4 \right) + u_{i,j+1}^{(k)}, \]
\[ u_{i+1,j}^{(k+1)} = \frac{1}{24} \left( S_1 + 2S_2 + 2S_3 + 7S_4 \right) + u_{i+1,j}^{(k)}. \]

By adding a weighted parameter \( r \) and \( \omega \) to equation (7), the implementation of the 4-EGAOR iterative method, introduced by Martins et al. [23], can be shown as

\[ u_{i,j}^{(k+1)} = \frac{1}{24} \left[ \omega \left( 7b_1 + s_2 + b_3 \right) + r \left( 7t_1 + 2t_2 \right) \right] + (1 - \omega) u_{i,j}^{(k)}, \]
\[ u_{i+1,j}^{(k+1)} = \frac{1}{24} \left[ \omega \left( 7b_2 + s_1 + b_3 \right) + r \left( 7c_4 + 2t_1 + c_3 \right) \right] + (1 - \omega) u_{i+1,j}^{(k)}, \]
\[ u_{i,j+1}^{(k+1)} = \frac{1}{24} \left[ \omega \left( 7b_3 + s_1 + b_2 \right) + r \left( 7c_3 + 2t_1 + c_4 \right) \right] + (1 - \omega) u_{i,j+1}^{(k)}, \]
\[ u_{i+1,j+1}^{(k+1)} = \frac{1}{24} \left[ \omega \left( 7b_4 + s_2 + b_1 \right) + r \left( 2t_2 + t_1 \right) \right] + (1 - \omega) u_{i+1,j+1}^{(k)}, \]

with

\[ b_1 = u_{i,j+1}^{(k+1)} + u_{i,j}^{(k+1)} - h^2 f_{i,j}, \quad b_2 = u_{i+1,j}^{(k+1)} + u_{i+1,j+1}^{(k+1)} - h^2 f_{i+1,j+1}, \]
\[ b_3 = u_{i+1,j+1}^{(k)} + u_{i+2,j+1}^{(k)} - h^2 f_{i+1,j+1}, \quad b_4 = u_{i+1,j+1}^{(k)} + u_{i+1,j+2}^{(k)} - h^2 f_{i+1,j+1}, \]
\[ c_1 = u_{i+1,j}^{(k)} - u_{i,j+1}^{(k)}, \quad c_2 = u_{i,j+1}^{(k+1)} - u_{i,j}^{(k)}, \quad c_3 = u_{i+1,j+1}^{(k+1)} - u_{i+1,j+1}^{(k)}, \quad c_4 = u_{i+1,j+1}^{(k+1)} - u_{i+1,j+1}^{(k)}, \]
\[ s_1 = 2(b_1 + b_3), \quad s_2 = 2(b_2 + b_4), \quad s_3 = 2(c_1 + c_2), \quad s_4 = 2(c_3 + c_4). \]

As shown in equation (7) and (8), with 4-EGAOR, the speed of computation is extremely improved since four node points are computed simultaneously in one loop of iteration. If there is no change of any node point from one loop to the next, the iterative process will terminated. The computation with very high precision is required to avoid the occurrence of flat region in the final solution. Direct methods will calculate the approximate values of the remaining node points once the temperature values of all black node points are obtained.

5. Experiments and Results
The experiment considered of three different sizes of static environment that consists different number of obstacles in various shapes of environment. In the initial setting, the walls and obstacles were fixed with high temperature values. The goal point was set to the lowest temperature values, while no initial value was assigned to the start point. All other free spaces were set to zero temperature value. The computation process was run on a PC running at 2.50GHz speed with 8GB of RAM. Until the stopping condition was met, iteration process to compute the temperature values numerically at all points continued. When there were no more changes in temperature values, where the difference between harmonic potentials at iterations \( k \) and \( k+1 \) was very small i.e. \( 1.0 \times 10^{-10} \), the loop would be terminated. In order to avoid the path generation solutions fails, this very high precision was necessary to avoid flat area, also known as saddle points. Tables 1 and 2 show the number of iterations and CPU time (in seconds) required to compute all temperature values in the environment for all numerical techniques compared in the experiments. Clearly, 4-EGAOR iterative method proved to be very fast compared to other considered methods.
Table 1. Performance of the considered methods in terms of number of iteration.

| Case | Methods | N x N | 300 x 300 | 600 x 600 | 900 x 900 |
|------|---------|-------|-----------|-----------|-----------|
| 1    | SOR     | 4748  | 16496     | 35574     |
|      | AOR     | 3899  | 13679     | 29605     |
|      | EGSOR   | 2446  | 8609      | 18772     |
|      | EGAOR   | 2008  | 7167      | 15627     |
| 2    | SOR     | 2885  | 10954     | 23515     |
|      | AOR     | 2363  | 9072      | 19546     |
|      | EGSOR   | 1471  | 5760      | 12331     |
|      | EGAOR   | 1186  | 4771      | 10263     |
| 3    | SOR     | 6066  | 23063     | 50473     |
|      | AOR     | 4992  | 19155     | 42056     |
|      | EGSOR   | 3134  | 12171     | 26607     |
|      | EGAOR   | 2587  | 10157     | 22180     |
| 4    | SOR     | 2058  | 7938      | 17553     |
|      | AOR     | 1668  | 6543      | 14391     |
|      | EGSOR   | 1042  | 4173      | 9131      |
|      | EGAOR   | 826   | 3441      | 7852      |

Table 2. Performance of the considered methods in terms of CPU time (in seconds).

| Case | Methods | N x N | 300 x 300 | 600 x 600 | 900 x 900 |
|------|---------|-------|-----------|-----------|-----------|
| 1    | SOR     | 19.30 | 394.30    | 1956.26   |
|      | AOR     | 18.40 | 364.28    | 1830.38   |
|      | EGSOR   | 11.58 | 232.50    | 1096.40   |
|      | EGAOR   | 9.97  | 203.67    | 969.42    |
| 2    | SOR     | 13.06 | 292.23    | 1284.88   |
|      | AOR     | 12.30 | 265.98    | 1159.02   |
|      | EGSOR   | 7.28  | 164.55    | 706.33    |
|      | EGAOR   | 6.47  | 142.17    | 602.96    |
| 3    | SOR     | 31.86 | 716.08    | 3200.68   |
|      | AOR     | 30.26 | 653.34    | 2784.98   |
|      | EGSOR   | 17.43 | 386.89    | 1645.57   |
|      | EGAOR   | 15.38 | 331.55    | 1406.37   |
| 4    | SOR     | 7.14  | 160.27    | 808.80    |
|      | AOR     | 6.56  | 144.02    | 661.77    |
|      | EGSOR   | 4.39  | 98.66     | 437.23    |
|      | EGAOR   | 3.94  | 86.59     | 386.35    |

Once the temperature values were obtained, the required path was generated by performing steepest descent search from start points to goal point. In all experiments, Figure 1 shows the paths in a known static environment that were successfully generated based on the Laplacian potential profile obtained via numerical computation. All start points (green dot/square dot) successfully ended at the specified goal point (red dot/circle dot), avoiding all various shapes of obstacles setting in the various environment. There’s some jagged nature of the paths due to the fact that no interpolation was performed. Interpolation of the gradient would provide smoother paths.

![Figure 1](image1.png)

**Figure 1.** The generated paths from several different start and goal positions for various environment.
6. Conclusion
The experiments in this study shows that solving Laplace’s equation (1) numerically to solve mobile robot path planning problem was indeed very attractive and feasible due to the recent advanced and new found numerical techniques, as well as the availability of faster machine todays. As shown in Table 1 and 2, the 4-EAGAOR iterative method proved to be very fast compared to the previous existing 4-EGGSOR method. An increase in the number of obstacles as well as various different shapes of obstacles does not affect the performance adversely, in fact the computation gets faster since the areas occupied by obstacles are ignored during computation. Apart from the concept of the full-sweep iteration, further investigation on half- [8,24,25] and quarter-sweep [26-29] iterations can also be considered in order to speed up the convergence rate of the proposed iterative method.

Acknowledgments
The authors acknowledge National Defence University of Malaysia for the completion of this article.

References
[1] Khatib O 1985 IEEE T Robotic Autom. 1 500-505
[2] Connolly C I, Burns J B and Weiss R 1990 Proc. IEEE Int. Conf. Robotic Autom. 2102-2106
[3] Akishita S, Hisanobu T and Kawamura S 1993 Proc. IEEE Int. Conf. Intelligent Robots Syst. 673-678
[4] Sasaki S 1998 Proc. IEEE Int. Conf. Control Appl. 1323-1327
[5] Karonava M, Zhelyazkov D, Todorova M, Penev I, Nikolov V and Petkov V Recent Res. Appl. Comp. Sci. 26-29
[6] Hachour O 2008 Int. J. Syst. Appl. Eng. Development 4(2) 178-190
[7] Connolly C I and Gruppen R 1993 J. Robotic Syst. 10(7) 931-946
[8] Saudi A and Sulaiman J 2013 Proc. Int. Conf. Technol. Adv. Electr. Electronics Comput. Eng. 424 – 429
[9] Saudi A, Sulaiman J and Hijazi M H A 2014 Res. J. Appl. Sci. 9(6) 354 – 360
[10] Saudi A and Sulaiman J 2012 Int. Symp. Robotics Intelligent Sensor 41 182 – 188
[11] Liang X, Wang H, Li D and Liu C 2014 Proc. IEEE Aerospace Conf. 1–13
[12] Motonaka K, Watanabe K and Maeyama S 2014 14th Int. Conf. Control Automat. Syst. (ICCAS) 1181–1184
[13] Pedersen M D and Fossen TI 2012 Proc. 9th IFAC Conf. Manoeuvring Control Mar. Craft. 188–193
[14] Shi C, Zhang M and Peng J 2007 7th Int. Conf. Telecommun. (ITST ’07) 1–6
[15] Masoud A A and Al-Shaikhi A 2015 54th IEEE Conf. Decision Control (CDC) 2761–2766
[16] Vallv´e J and Andrade-Cetto J 2015 Robotic Auton. Syst. 69 68–79
[17] Evans L C 1998 Partial Differential Equations Providence: American Mathematical Society
[18] Ibrahim A 1993 The study of the iterative solution of boundary value problem by the finite difference method PhD thesis, Universiti Kebangsaan Malaysia (UKM)
[19] Sulaiman J, Hasan M K and Othman M 2007 in O. Gervasi & M. Gavrilova (Eds). Comput. Sci. Appl. 2007. Lecture notes in Comp. Sci. (LNCS 4707): 298-308. Berlin: Springer-Verlag
[20] Evans D J 1985 Int. J. Comp. Maths. 17 81-108
[21] Evans D J and Yousif W S 1986 Int. J. Comp. Maths. 18 323-340
[22] Hadjidimos A 1978 Math. Comput. 32(141) 149-157
[23] Martins M M, Yousif W S and Evans D J 2002 Neural, Parallel and Science Computation 10(4) 411-422
[24] Dahalan A A and Sulaiman J 2016 Appl. Math. Sci. 10(2) 45-57
[25] Dahalan A A, Shattar N A and Sulaiman J 2016 J. Eng. Appl. Sciences 11(9) 1891-1897
[26] Dahalan A A and Sulaiman J 2015 Int. J. Math. Anal. 35(9) 1733-1746
[27] Dahalan A A, Aziz N S A and Sulaiman J 2016 J. Eng. Appl. Sciences 11(7) 1456-1463
[28] Sulaiman J, Othman M and Hassan M K 2009 Chamchuri J. Math. 2(1) 93-103
[29] Muthuvalu M S and Sulaiman J 2010 Appl. Math. Sci. 4(59) 2943-2953