Jurnal Teknologi

Full Paper

PATH PLANNING SIMULATION USING HARMONIC POTENTIAL FIELDS THROUGH FOUR POINT-EDGSOR METHOD VIA 9-POINT LAPLACIAN

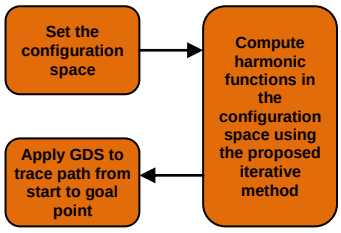
Azali Saudi[a*], Jumat Sulaiman[b]

[a]Faculty of Computing and Informatics, Universiti Malaysia Sabah, Kota Kinabalu, Malaysia
[b]Faculty of Science and Natural Resources, Universiti Malaysia Sabah, Kota Kinabalu, Malaysia

Article history
Received
13 October 2015
Received in revised form
31 March 2016
Accepted
28 February 2016

*Corresponding author
azali@ums.edu.my

Graphical abstract



Abstract

This paper presents our study on a simulation of path planning for indoor robot that relies on the use of Laplace's equation to constrain the generation of Harmonic Potential Fields (HPF). The computation of HPF requires immense amount of computing resources, particularly when the size of environment is large. In the past, fast iterative methods that apply the use of half-sweep iteration and block technique are suggested. In this study, faster iterative method known as Four Point-Explicit Decoupled Group Successive Over relaxation via 9-Point Laplacian (4-EDGSOR-9L) is introduced. Essentially, the 4-EDGSOR-9L is actually a variant of block SOR iterative method based on four points that employs half-sweep iteration and utilizes 9-Point Laplacian discretization scheme. Once the HPF is obtained, the standard Gradient Descent Search (GDS) technique is performed for path tracing to the goal point.

Keywords: Path planning simulation; Explicit Decoupled Group SOR; iterative method

Abstrak

Kertas kerja ini menerangkan kajian tentang simulasi perancangan laluan suatu robot yang menggunakan persamaan Laplace untuk menjana *Harmonic Potential Fields (HPF)*. Pengiraan *HPF* memerlukan sumber komputer yang banyak, terutamanya apabila ia melibatkan saiz persekitaran yang besar. Dalam kajian-kajian yang lepas, kaedah lelaran pantas yang menggunakan lelaran sapuan-separuh dan teknik blok telah dicadangkan. Dalam kajian ini, kaedah lelaran lebih pantas yang dinamakan sebagai *Four Point-Explicit Decoupled Group Successive Overrelaxation via 9-Point Laplacian (4-EDGSOR-9L)* diperkenalkan. Sebenarnya, kaedah lelaran *4-EDGSOR-9L* adalah variasi blok *SOR* berasaskan empat titik yang menggunakan lelaran sapuan-separuh dan skema pendiskretan 9-titik Laplacian. Setelah HPF diperolehi, teknik lazim *Gradient Descent Search (GDS)* digunakan untuk menjejak laluan ke titik destinasi.

Kata kunci: Simulasi perancangan laluan; *Explicit Decoupled Group SOR;* kaedah lelaran

1.0 INTRODUCTION

A truly autonomous robot must have the capability to find path from its start point to a specified goal point. This study proposed a robot path planning technique that relies on the use of Laplace's equation to constrain the generation of Harmonic Potential Fields (HPF). The application of HPF to solve path planning problem was first demonstrated by Connolly et al. [1]. In the past, computing HPF using numerical techniques

produced encouraging results. The standard numerical implementations include approaches based on family of point iterative methods i.e. Jacobi, Gauss-Seidel and Successive Over relaxation (SOR) [1-4]. Additionally, fast iterative methods that utilize half-sweep iteration and block computation technique were also employed in our previous works [5-10]. The main challenge is to improve the computation speed of obtaining the HPF, since it often requires immense amount of computing resources when the size of environment is large. Hence, in this study, faster iterative method known as Four Point-Explicit Decoupled Group SOR via 9-Point Laplacian (4-EDGSOR-9L) is introduced for computing HPF to further improve the overall performance of the path planning algorithm. Once the HPF are obtained, the standard Gradient Descent Search (GDS) technique is performed for quick path tracing from start point to the goal point.

2.0 LITERATURE REVIEW

The HPF is actually the solutions of Laplace's equation, and they have been typically used as a global method for robot path planning [1,2,4,11], where it were shown that the HPF has a number of properties which are essential to robotics applications. Paths derived from HPF are generally smooth and they also offer a complete path planning algorithm. They are guaranteed to always provide a trajectory to the goal by following a path of gradient descent and therefore, do not suffer from problems of local minima. Consequently, they can be used to advantage for potential-field path planning. However, a solution may be computationally expensive and gradient values may be small and indistinguishable from noise. Alternatively, to circumvent these issues, it is argued that HPF can be used for complete local 2-D path planning provided that its spatial extent is limited and it is integrated with a global path planner to compensate for the imposed myopia [12].

In the literature, Khatib [13] introduced the use of potential fields for robot path planning. It views every obstacle to be exerting a repelling force on an end effector, while the goal exerts an attractive force. Koditschek [14], using geometrical arguments, showed that, at least in certain types of domains, there exists potential fields which can guide the effect or from almost any point to a given point. These potential fields forpath planning, however, suffer from the spontaneous creation of local minima. Meanwhile, global path planning using Laplace's equation was first introduced in the pioneer work by Connolly et al. [1] and Akishita et al. [2]. Exact robot navigation using artificial potential fields was developed by Rimon and Koditschek [15]. HPF was employed for real-time obstacle in [16]. It was observed by Connolly and Grupen [11] that HPF had a number of properties useful in robotic applications. The work by Sasaki [4] demonstrated the use of numerical technique for solving path planning problem. Then, Waydo and Murray [17] utilized stream functions that are similar to HPF to generate motion planning for a vehicle. Path planning using HPF and probabilistic cell decomposition was carried out by Rosell and Iniguez [18]. Daily and Bevly [19] used HPF for path planning of high speed vehicles. Meanwhile, finite elements had been applied to obtain HPF for robotic motion [20]. Morerecently, Szulczynski et al. [21] used HPF for real-time obstacles avoidance, whilst Yang and Ariyur [22] implemented Laplacian path planning for avoiding moving obstacles. Pedersen and Fossen [23] applied a potential flow solver to marine vessel path planning for cluttered environments. A three dimensional (3D) potential path planning method for unmanned aerial vehicles (UAVs) in complex environments is studied by Liang et al. [24].

Although, there are many elegant analytical solutions to Laplace's equation in special geometries, the practical real problems are usually solved numerically. The availability of modern and powerful computers and software made it easier to obtain the numerical solutions of Partial Differential Equations. By using finite difference method for solving the Laplace's equation, the Partial Differential Equation is converted into a set of linear simultaneous equations i.e. linear system. Generally, methods for solving linear system such as Laplace's equation can be classified into two main group i.e. direct and iterative methods. Essentially, direct methods are recommended for linear system with dense and unstructured coefficient matrix, whereas iterative methods are best for very large sparse matrices [25].Iterative methods offer a vast saving of storage space compared with direct methods, since usually only the non-vanishing elements of the system matrix, the solution vector, and a few additional vectors have to be stored. However, one of the disadvantages of iterative methods compared with direct methods is slow convergence or even divergence [26].

In recent decades, complexity reduction approach has been applied vigorously for computing the solutions of linear systems such as Poisson and Laplace's equations. The basic idea of complexity reduction approach such as half-sweep iteration is to reduce the computational complexity of the solution methods. The half-sweep iteration concept isfirst envisioned via the Explicit Decoupled Group (EDG) [27] method for solving Poisson equation. The EDG method is actually an extension from the standard Explicit Group (EG) [28] method, whilst its faster variant that employs weighted parameter via SOR is known as EDGSOR method. Since then, the EDG and EDGSOR methods had been employed for solving linear systems generated from various problems [29-31]. Also, half-sweep iteration technique and its variants were utilized in many iterative methods for computing the solutions of linear systems [32-35].

3.0 PATH PLANNING STRATEGY

Essentially, the aim of robot path planning is to construct a collision-free path from some initial configurations to some goal configurations for a robot within a workspace containing obstacles. This study is inspired by the physical analogy of heat transfer in which path planning strategy is constructed by utilizing the temperature distributions model in the environment of the robot as described in Section 3.1. The mathematical model of the path planning problem relies on the solutions of Laplace's equation, i.e. harmonic functions, to provide surface gradients that are useful for robot navigation purposes [11].In robotics, these harmonic functions are also known as Harmonic Potential Fields (HPF). Harmonic functions are discussed in detail in Section 3.2. The configuration spaces are constructed based on the previous work [1] so that better performance comparison can be obtained. Section 3.3 described the details of the configuration space used in this study. Once the temperature distributions of the environment are obtained, the path can be generated by using the temperature gradient to navigate the configuration space as explained in Section 3.4. The self-developed robot simulator used in this study is described in Section 3.5.

3.1 Physical Analogy

Assume that a real robot vehicle can be reduced to a point moving in a known environment, path planning problem of the robot can be formulated as a steady-state heat transfer problem. In the heat transfer analogy, the goal is treated as a sink pulling heat in. Whilst, the environment boundaries and obstacles are considered as heat sources and are fixed with constant temperature values. As a result of heat conduction process, the temperature distributions develop and the heat flux lines that are flowing to the sink fill the work space. The path then can be easily found by following the heat flux.

Based on the above analogy, a global path planning method can be developed by using Laplace's equation to model the temperature distributions in the environment. These temperature distributions represent the HPF of the environment. The HPF is computed in global manner over the entire region of the environment, and is used to find path lines for a robot to move from the start point to the goal point. Outer boundaries, inner walls and obstacles are considered as current heat sources to be assigned with high fixed potential, whereas the goal is considered to be the sink with the lowest fixed assigned potential. By using iterative method, the HPF is computed iteratively until the convergence criterion is satisfied. Then, by performing the gradient descent strategy on the computed HPF, a path is generated by following the current line and move to succession of points with lower potential leading to the goal point with the least potential[36].

3.2 Harmonic Functions

Harmonic functions are actually solutions to Laplace's equation [37]. They are known to have a number of properties useful for robotics applications [11]. Harmonic functions offer a complete path planning algorithm and paths derived from them are generally smooth. One main advantage of harmonic functions, when applied to robot path planning, is that they exhibit no spurious local minima.

Harmonic functions are functions which satisfies Laplace's equation,

$$\nabla^2 \emptyset = \sum_{i=1}^{n} \frac{\partial^2 \emptyset}{\partial x_i^2} = 0 \tag{1}$$

where x_i is the i-th Cartesian coordinate and n is the dimension. In the case of robot path construction, the boundary consists of the outer boundary walls of the environment and inner walls of all obstacles. The spontaneous creation of a false local minimum inside the region is avoided if Laplace's equation is imposed as a constraint on the functions used, as the harmonic functions satisfy the min-max principle [38].The gradient vector field of a harmonic function has a zero curl, and the function itself obeys the min-max principle. Hence the only types of critical points which can occur are saddle points or flat regions. For a path-planning algorithm, an escape from such critical points can be found by performing a search in the neighbourhood of that point. Moreover, any perturbation of a path from such point results in a path which is smoothes everywhere[11].

Though harmonic functions are free from local minima, they are not fully global. In path planning problem, Koditschek [14] showed that in the presence of obstacles, a global avigation function does not exist in general. For a two dimensional space with q disjoint obstacles, a potential function U must possess at leastq saddle points. In general, harmonic functions give rise to a practical path planning approach with the following main features [39]: i) a potential field with a unique minimum, ii) an efficient update of the potential field, iii) completeness up to the discretization error in the environment, and iv) a robust and reactive behavior.

3.3 Configuration Space

In the framework used in this study, the robot is represented by a point in the configuration space. The path planning problem is then posed as an obstacle avoidance problem for the point robot from the start to the goal point in the configuration space. The configuration space has outer boundaries and some obstacles inside the boundary. The configuration space is designed in grid form and the coordinates and potential values associated with each node are computed iteratively to satisfy equation (1).The solutions to Laplace's equation are subjected to Dirichlet boundary conditions where a constant value

is given at each point of the boundary. Thus, high fixed potential value is assigned to the outer boundaries, inner walls and obstacles, and lowest fixed potential value for the goal point. No initial potential values are assigned to all other free non-occupied points and the start points. Hence, by using the heat transfer analogy as described in Section 3.1, the boundary of the configuration space is modeled as a source, and the boundary of the goal is modeled as a sink. The computation of HPF only involves free non-occupied points, since all points occupied by obstacles are ignored during the iteration process.

The configuration space samples are taken from the pioneer work by Connolly et al.[1]. The original samples are in varying sizes i.e. 200 by 200 grid, 30 by 41, 50 by50 and 70 by 70. All these samples are rescaled into grid of 300 by 300, so that better performance comparisons can be made. In the literature, the numerical experiments were conducted against several mesh sizes in order to obtain better performance comparisons [30-35]. Therefore, the considered sizes of the configuration space are grids of 300 by 300, 600 by 600, 900 by 900, 1200by 1200, 1500 by 1500 and 1800 by 1800.

3.4 Path Generation

Once the harmonic functions under the boundary conditions are established using the iterative methods as further described in Section 4.0, the required path can be traced by using standard GDS searching technique [1,4,12,20,36]. GDS employs simple technique by following the negative gradient from the start point through successive points with lower temperature till the goal, which is the point with the lowest temperature.

3.5 Robot Simulator

The self-developed robot simulator software is developed using Lazarus, and the code is written in Object-Pascal. Commercial robot simulators are available such as WEBOTS [40] and Open Sim, but this self-developed simulator provides maximum flexibility, albeit with limited features.

4.0 THE ITERATIVE METHODS

The solutions to Laplace's equation are called harmonic functions (also known as HPF in robotics). Considering a 2D configuration space, the HPF can be expressed in a 2D version of Laplace's equation as

$$\nabla^2 f = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{2}$$

The operator ∇^2 is called the Laplacian. The Laplace's equation can be solved in one of two ways, analytically [19] or numerically [1,3,12]. In this study, the solutions of Laplace's equation are obtained using numerical method particularly the finite difference method. In this method, the Laplace's equation is converted into a set of linear simultaneous equations (or linear system). When the linear system is written in matrix notation, the majority of the elements of the matrix are zero. Such matrices are called sparse matrix. In the case of the mathematical model of path planning problem, the resulting linear system becomes very large and sparse, thus requiring very large storage in memory of the computer. Therefore in this study, by following the suggestion in [1], a more efficient way of solving this very large linear system using iterative method is employed. The main advantage of iterative solution is that the storing of large matrices is unnecessary.

In order to develop accurate and efficient numerical iterative methods for solving problem(2), the solution needs to be discretized in a suitable way so that it can be stored in a computer. The standard finite difference discretization technique is based on 5-Point Laplacian (5L). In the previous studies [5-7], the point Full-Sweep SOR (FSSOR) and Half-Sweep SOR (HSSOR)methods, and the block method namely Four Point-Explicit Group SOR (4-EGSOR) that based on 5L were successfully applied to solve problem (2). The block variant of half-sweep SOR iteration namely Four Point-Explicit Decoupled Group SOR (4-EDGSOR) will be considered in this study. The FSSOR and HSSOR methods, and their corresponding block variantsi.e. the 4-EGSOR and 4-EDGSOR methods, are described in Sections 4.3 and 4.4, respectively.

Additionally, iterative methods based on 9-Point Laplacian (9L)i.e. the point FSSOR-9L method and its block variant 4-EGSOR-9L method, were employed previously as reported in [8,9]. Also, the point HSSOR-9L method that apply half-sweep iteration technique via 9L was implemented in [10]. The FSSOR-9L and HSSOR-9L methods are further described in Section 4.5, whilst the 4-EGSOR-9L method is described in Section 4.6. In this study, the block variant of half-sweep SOR iteration via 9L method, namely the 4-EDGSOR-9L method, as further described in Section 4.7, is examined to further improve the overall performance of the path planning algorithm.

4.1 The 5-Point Laplacian (5L)

To discuss discretization, first consider the Laplacian in 2D given by

$$\nabla^2 f(x,y) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \tag{3}$$

The 2D Laplacian then can be approximated using the five-point stencil finite difference method to obtain:

$$\nabla^2 f(x,y) \approx \frac{u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h) - 4u(x,y)}{h^2} \tag{4}$$

where $f(x,y)$ is a function which satisfies Laplace's equation, $u(x,y)$ represents a discrete regular sampling of f on a grid, and h is the step size to be used in approximating thederivatives in each direction. The equation (4) is second order accurate because the

error is of the order of h^2. It is derived from Taylor series approximation to the second derivatives using central difference formula at x and y directions:

$$u(x+h,y) = u(x,y) + h\frac{\partial u}{\partial x} + \frac{1}{2}h^2\frac{\partial^2 u}{\partial x^2} + \frac{1}{6}h^3\frac{\partial^3 u}{\partial x^3} + O(h^4) \tag{5}$$

$$u(x-h,y) = u(x,y) - h\frac{\partial u}{\partial x} + \frac{1}{2}h^2\frac{\partial^2 u}{\partial x^2} - \frac{1}{6}h^3\frac{\partial^3 u}{\partial x^3} + O(h^4) \tag{6}$$

Then add both equations (5) and (6) together to get the following equations along the x and y directions, respectively:

$$u(x+h,y) + u(x-h,y) = 2u(x,y) + h^2\frac{\partial^2 u}{\partial x^2} + O(h^4) \tag{7}$$

$$u(x,y+h) + u(x,y-h) = 2u(x,y) + h^2\frac{\partial^2 u}{\partial y^2} + O(h^4) \tag{8}$$

Then combine both equations (7) and (8), thus the following equation is obtained:

$$u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h) = 4u(x,y) + h^2\frac{\partial^2 u}{\partial x^2} + h^2\frac{\partial^2 u}{\partial y^2} + O(h^4) \tag{9}$$

If the fourth order error terms are discarded, and the step sizes are all equal, the following second-order central difference approximation to the second derivative is obtained:

$$\nabla^2 f(x,y) = \frac{1}{h^2}[u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h) - 4u(x,y)] \tag{10}$$

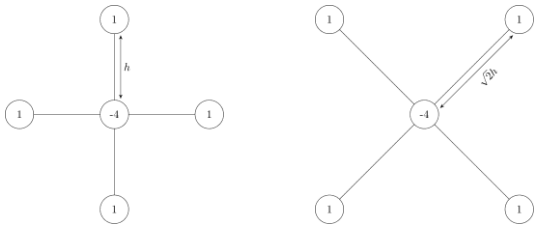
Another type of approximation is based on the cross orientation operator which can be obtained by rotating the x-y axis 45°[29]. This will result in the rotated (skewed) 5L approximation and be written as

$$\nabla^2 f(x,y) = \frac{1}{2h^2}[u(x-h,y-h) + u(x+h,y-h) + u(x-h,y+h) + u(x+h,y+h) - 4u(x,y)] \tag{11}$$

Essentially, the 5L approximations in equations (10) and (11) represent the full-sweep and half-sweep iteration cases, respectively. The computational molecules for the corresponding 5L approximations for both full-sweep and half-sweep iterations are shown in Figure 1 [27]. Whilst, Figure 2 shows the portion of the computational grid for the 5L about point (i,j) for full-sweep and half-sweep cases[27].The 5L approximations for full-sweep and half-sweep cases can also be written in stencil forms as shown in equations (12) [41] and (13) [42], respectively.

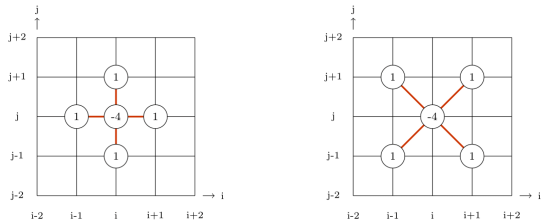
$$\nabla^2 f = \frac{1}{h^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{12}$$

$$\nabla^2 f = \frac{1}{2h^2}\begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix} \tag{13}$$



(a) (b)

Figure 1 The computational molecules of the 5L approximations for (a) full-sweep and (b) half-sweep cases



(a) (b)

Figure 2 Portion of the computational grid for the 5L about point (i, j) for (a) full-sweep and (b) half-sweep cases

Based on the 5L approximations in equations (10) and (11), let $U_{i,j}$ representsan approximation to $f(x,y)$. Then, the full-sweep and half-sweep approximation equations for problem (2) can be rewritten as

$$U_{i-1,j} + U_{i+1,j} + U_{i,j-1} + U_{i,j+1} - 4U_{i,j} = 0 \tag{14}$$

and

$$U_{i+1,j+1} + U_{i-1,j-1} + U_{i+1,j-1} + U_{i-1,j+1} - 4U_{i,j} = 0, \tag{15}$$

respectively. Now, by applying these finite difference approximations to problem (2), it will result in a large and sparse linear system that can be stated in matrix form as

$$Au = b \tag{16}$$

where the matrix A and the column vector b are both known, and the column vector u isunknown.Since the linear system in equation (16) is large and sparse, the iterative method is suitable to solve this type of problem. Also it can be solved either by point or block iterative methods [27]. Thus, the Gauss-Seidel iterative schemes for the full-sweep and half-sweep cases on

the finite difference equations (14) and (15) can be constructed and are given as

$$U_{i,j}^{(k+1)} = \frac{1}{4}\left[U_{i-1,j}^{(k+1)} + U_{i+1,j}^{(k)} + U_{i,j-1}^{(k+1)} + U_{i,j+1}^{(k)}\right] \tag{17}$$

and

$$U_{i,j}^{(k+1)} = \frac{1}{4}\left[U_{i+1,j+1}^{(k)} + U_{i-1,j-1}^{(k+1)} + U_{i+1,j-1}^{(k+1)} + U_{i-1,j+1}^{(k)}\right], \tag{18}$$

respectively. Furthermore, by adding a weighted parameterω, via SOR [43], the SOR iterative schemes for the full-sweep and half-sweep cases are given as

$$U_{i,j}^{(k+1)} = \frac{\omega}{4}\left[U_{i-1,j}^{(k+1)} + U_{i+1,j}^{(k)} + U_{i,j-1}^{(k+1)} + U_{i,j+1}^{(k)}\right] + (1-\omega)U_{i,j}^{(k)} \tag{19}$$

and

$$U_{i,j}^{(k+1)} = \frac{\omega}{4}\left[U_{i+1,j+1}^{(k)} + U_{i-1,j-1}^{(k+1)} + U_{i+1,j-1}^{(k+1)} + U_{i-1,j+1}^{(k)}\right] + (1-\omega)U_{i,j}^{(k)}, \tag{20}$$

respectively. Generally, the optimal value ofω is defined in the range $0 < \omega < 2$[43]. Note that if $\omega = 1$, the SOR method simplifies to the standard Gauss-Seidel method. Section 4.3 presents the application of equations (19) and (20) to develop the FSSOR and HSSOR iterative methods for computing the solutions of Laplace's equation.

4.2 The 9-Point Laplacian (9L)

Another possible approximation of the 2D Laplacian (3.1) is the 9L [37]

$$\nabla^2 f(x,y) = \frac{1}{6h^2}[4u(x-h,y) + 4u(x+h,y) + 4u(x,y-h) + 4u(x,y+h) + u(x-h,y-h) + u(x+h,y-h) + u(x-h,y+h) + u(x+h,y+h) - 20u(x,y)] \tag{21}$$

In comparison to the 5L approximation, the 9L approximation utilizes 9 computational molecules as shown in Figure 3, thus it produces more accurate solution. Furthermore, by rotating the x-y axis 45°, the rotated 9L approximation is obtained and can be written as[44]

$$\nabla^2 f(x,y) = \frac{1}{12h^2}[4u(x+h,y-h) + 4u(x+h,y+h) + 4u(x-h,y+h) + 4u(x-h,y-h) + u(x+2h,y) + u(x,y+2h) + u(x-2h,y) + u(x,y-2h) - 20u(x,y)] \tag{22}$$

The 9L approximations as given in equations (21) and (22) represent the full-sweep and half-sweep iterations, respectively. Figures 3 and 4 illustrate the computational molecules and the portion of computational grid for full-sweep and half-sweep cases, respectively. The 9L approximations for full-sweep and half-sweep cases can also be written in stencil form as [44]

$$\nabla^2 f = \frac{1}{h^2}\begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix} \tag{23}$$

and

$$\nabla^2 f = \frac{1}{h^2}\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 4 & 0 & 4 & 0 \\ 1 & 0 & -20 & 0 & 1 \\ 0 & 4 & 0 & 4 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}, \tag{24}$$

respectively. Let $U_{i,j}$ represents an approximation to $f(x,y)$. By considering the 9L approximations in equations (21) and (22), the full-sweep and half-sweep approximation equations for problem (2) can be rewritten as

$$4(U_{i-1,j} + U_{i+1,j} + U_{i,j-1} + U_{i,j+1}) + U_{i-1,j-1} + U_{i+1,j-1} + U_{i-1,j+1} + U_{i+1,j+1} - 20U_{i,j} = 0 \tag{25}$$

and

$$4(U_{i+1,j-1} + U_{i+1,j+1} + U_{i-1,j+1} + U_{i-1,j-1}) + U_{i+2,j} + U_{i,j+2} + U_{i-2,j} + U_{i,j-2} - 20U_{i,j} = 0, \tag{26}$$

respectively. Similarly, when these finite difference approximations are applied to equation (2), it will generate a linear system that is very large and sparse. By solving this linear system iteratively, the iterative schemes for the full-sweep and half-sweep cases on the finite difference equations (25) and (26) can be defined as

$$U_{i,j}^{(k+1)} = \frac{1}{4}\left[U_{i-1,j}^{(k+1)} + U_{i+1,j}^{(k)} + U_{i,j-1}^{(k+1)} + U_{i,j+1}^{(k)}\right] \tag{27}$$

and

$$U_{i,j}^{(k+1)} = \frac{1}{4}\left[U_{i+1,j+1}^{(k)} + U_{i-1,j-1}^{(k+1)} + U_{i+1,j-1}^{(k+1)} + U_{i-1,j+1}^{(k)}\right], \tag{28}$$

respectively. Furthermore, by using an accelerated parameterω, the corresponding SOR iterative schemes for the full-sweep and half-sweep methods via 9L are given as

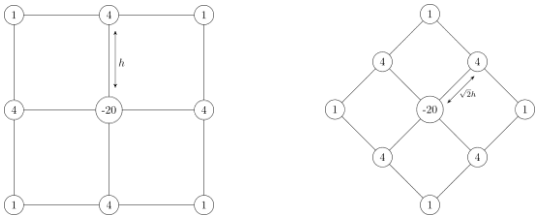
$$U_{i,j}^{(k+1)} = \frac{\omega}{4}\left[U_{i-1,j}^{(k+1)} + U_{i+1,j}^{(k)} + U_{i,j-1}^{(k+1)} + U_{i,j+1}^{(k)}\right] + (1-\omega)U_{i,j}^{(k)} \tag{29}$$

and

$$U_{i,j}^{(k+1)} = \frac{\omega}{4}\left[U_{i+1,j+1}^{(k)} + U_{i-1,j-1}^{(k+1)} + U_{i+1,j-1}^{(k+1)} + U_{i-1,j+1}^{(k)}\right] + (1-\omega)U_{i,j}^{(k)}, \tag{30}$$

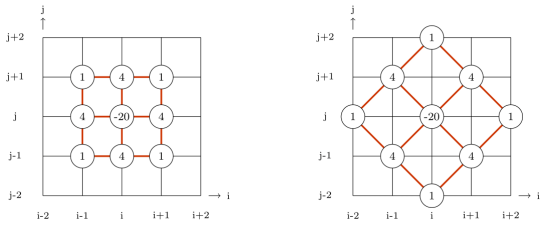
respectively. Section 4.5 presents the application of equations (29) and (30) to develop the algorithms

ofFSSOR-9L and HSSOR-9L methods for computing the solutions of Laplace's equation.



(a) (b)

Figure 3 The computational molecules of the 9L approximations for (a) full-sweep and (b) half-sweep cases



(a) (b)

Figure 4 Portion of the computational grid of the 9L about point (i, j) for (a) full-sweep and (b) half-sweep cases

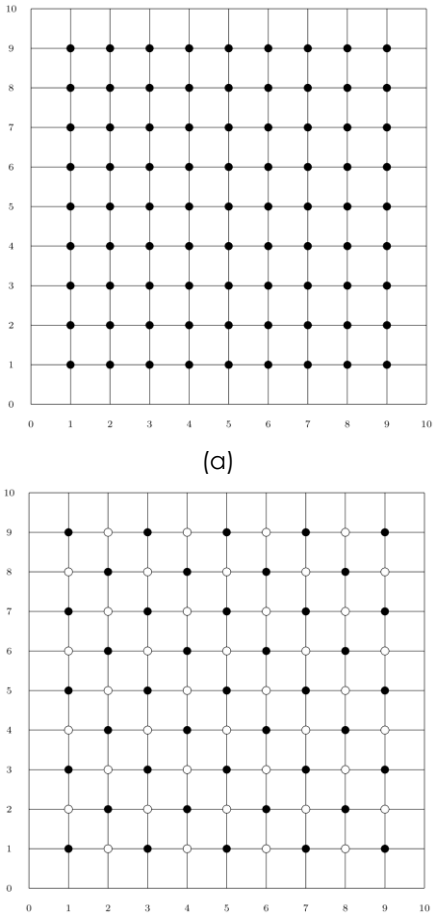
4.3 The FSSOR and HSSOR methods

The FSSOR is actually the standard SOR [43] method that employs traditional full-sweep iteration using five-point discretization scheme. Whilst, the HSSOR [27] method is formulated by utilizing half-sweep iteration concept, and also using five-point discretization scheme. The algorithms for the implementation of FSSOR and HSSOR methods utilize the SOR iterative schemes as given in equations (19) and (20), respectively.

With FSSOR method, all nodes are computed during the iteration process. Whilst, the HSSOR method utilizes half-sweep iteration, thus only half of all nodes are computed, since only black nodes are considered during the iteration process until the convergence criterion is reached. The remaining white nodes are then computed using direct technique [27]. Hence, the computational complexity of HSSOR method is reduced by approximately half. Figure 5 illustrates the computational nodes for full-sweep and half-sweep iterations.

4.4 The 4-EGSOR and 4-EDGSOR methods

This section describes the block variants of the FSSOR and HSSOR methods namely the 4-EGSOR and 4-EDGSOR methods, respectively. Let us consider a group of four points as illustrated in Figure 6. By considering the standard 5Lapproximation in equation (14), the 4-EGSOR method can be generally expressed as [27,28]



(a)

(b)

Figure 5 Computational nodes for (a) full-sweep and (b) half-sweep cases

$$\begin{bmatrix} 4 & -1 & -1 & 0 \\ -1 & 4 & 0 & -1 \\ -1 & 0 & 4 & -1 \\ 0 & -1 & -1 & 4 \end{bmatrix} \begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \tag{31}$$

where

$$\begin{aligned} S_1 &= U_{i-1,j} + U_{i,j-1}, \\ S_2 &= U_{i+2,j} + U_{i+1,j-1}, \\ S_3 &= U_{i-1,j+1} + U_{i,j+2}, \\ S_4 &= U_{i+2,j+1} + U_{i+1,j+2}. \end{aligned}$$

By determining the inverse of coefficient matrix, equation (31) can be rewritten as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix} = \frac{1}{24} \begin{bmatrix} 6S_1 + S_a \\ 6S_2 + S_b \\ 6S_3 + S_b \\ 6S_4 + S_a \end{bmatrix} \tag{33}$$

where

$$\begin{aligned} S_a &= 2(S_2 + S_3) + S_1 + S_4, \\ S_b &= 2(S_1 + S_4) + S_2 + S_3. \end{aligned}$$

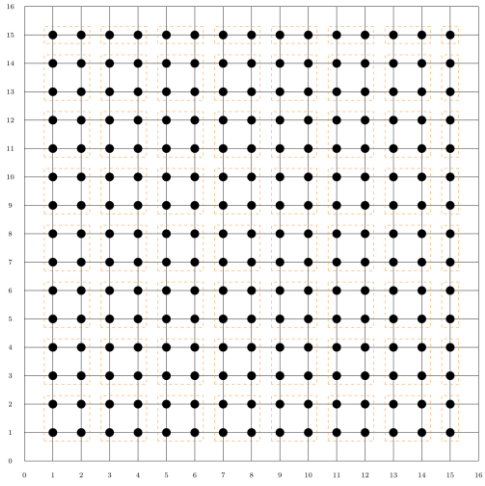


Figure 6 Computational nodes for 4-EGSOR method

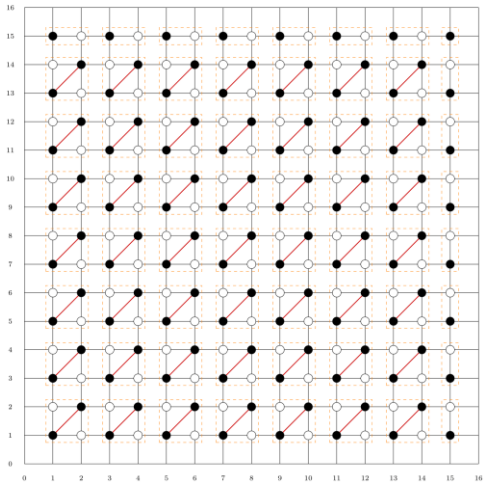


Figure 7 Computational nodes for 4-EDGSOR method

Now, the SOR iterative scheme for equation (33) can be written as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix}^{(k+1)} = \frac{\omega}{24}\begin{bmatrix} 6S_1 + S_a \\ 6S_2 + S_b \\ 6S_3 + S_b \\ 6S_4 + S_a \end{bmatrix} + (1-\omega)\begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix}^{(k)} \tag{34}$$

By employing block technique, the 4-EGSOR method obtains four potential values per computation. For the computation of groups of points near to boundary, they can be treated as groups of two points and single point, as shown in Figure 6. Thus, for computation of these groups, direct method is employed using equation (17) [28].

Similarly, for 4-EDGSOR method, let a group of four points (see Figure 7) be considered based on the rotated 5L approximation equation (15) to form a(4x4) system of linear algebraic equations and be given as [27,30,31]

$$\begin{bmatrix} 4 & -1 & 0 & 0 \\ -1 & 4 & 0 & 0 \\ 0 & 0 & 4 & -1 \\ 0 & 0 & -1 & 4 \end{bmatrix}\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \\ U_{i+1,j} \\ U_{i,j+1} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \tag{35}$$

where

$$\begin{aligned} S_1 &= U_{i-1,j-1} + U_{i-1,j+1} + U_{i+1,j-1}, \\ S_2 &= U_{i,j+2} + U_{i+2,j+2} + U_{i+2,j}, \end{aligned}$$

and

$$\begin{aligned} S_3 &= U_{i,j-1} + U_{i+2,j-1} + U_{i+2,j+1}, \\ S_4 &= U_{i-1,j} + U_{i-1,j+2} + U_{i+1,j+2}. \end{aligned}$$

The linear system (35) is then decomposed independently into two (2x2) linear algebraic equations as

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \tag{36}$$

and

$$\begin{bmatrix} 4 & -1 \\ -1 & 4 \end{bmatrix}\begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix} = \begin{bmatrix} S_3 \\ S_4 \end{bmatrix}. \tag{37}$$

By determining the inverse of coefficient matrix, equations (36) and (37) can be rewritten as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix} = \frac{1}{15}\begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix}\begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \tag{38}$$

and

$$\begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix} = \frac{1}{15}\begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix}\begin{bmatrix} S_3 \\ S_4 \end{bmatrix}, \tag{39}$$

respectively. Accordingly, the SOR iterative scheme for equations (38) and (39) can be written independently as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix}^{(k+1)} = \frac{\omega}{15}\begin{bmatrix} 4S_1 + S_2 \\ S_1 + 4S_2 \end{bmatrix} + (1-\omega)\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix}^{(k)} \tag{40}$$

and

$$\begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix}^{(k+1)} = \frac{\omega}{15}\begin{bmatrix} 4S_3 + S_4 \\ S_3 + 4S_4 \end{bmatrix} + (1-\omega)\begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix}^{(k)} \tag{41}$$

respectively. With 4-EDGSOR method, the configuration space consists of black and white nodes, as illustrated in Figure 7. The computations are carried out on groups of either pair of black or white nodes only. For the computation of groups of nodes near to boundary, i.e. groups with only one or two nodes, direct method is employed by using equation (18) [28]. Hence, the implementation of 4-EDGSOR method may utilize either equation (40) or (41). After the convergence, all the remaining white nodes are evaluated by applying direct method using equation (17).

4.5 The FSSOR-9L and HSSOR-9L methods

The FSSOR-9L and HSSOR-9L methods utilize the SOR iterative schemes via 9L for full-sweep and half-sweep cases as given in equations (29) and (30), respectively. The FSSOR-9L method considers all nodes in the configuration space. Whilst, the more efficient HSSOR-9L method considers only half of the total nodes, since only black nodes are considered during the iteration process, as illustrated in Figure 8. Once the convergence of the iteration process is reached, the remaining white nodes are then computed using direct technique [27].

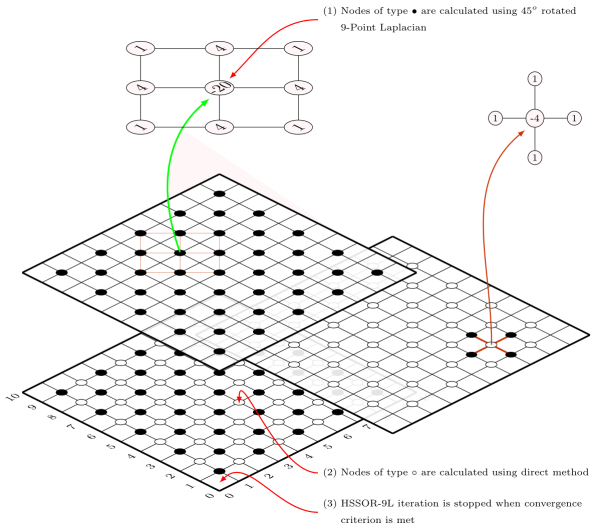


Figure 8 HSSOR-9L method considers only half of the total nodes during the iteration process, whilst the remaining nodes are computed using direct method after the convergence is reached

4.6 The 4-EGSOR-9L method

The block variant of full-sweep iteration via 9L namely 4-EGSOR-9L method utilizes the SOR iterative scheme via 9L as given in equation (29). Thus, to formulate the 4-EGSOR-9L iterative method, let us consider a group of four points as illustrated in Figure 9 and defined as[45]

$$\begin{bmatrix} 20 & -4 & -4 & -1 \\ -4 & 20 & -1 & -4 \\ -4 & -1 & 20 & -4 \\ 0 & -4 & -4 & 20 \end{bmatrix} \begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \tag{42}$$

where

$$\begin{aligned} S_1 &= 4U_{i-1,j} + 4U_{i,j-1} + U_{i-1,j-1} + U_{i+1,j-1} + U_{i-1,j+1}, \\ S_2 &= 4U_{i+2,j} + 4U_{i+1,j-1} + U_{i,j-1} + U_{i+2,j-1} + U_{i+2,j+1}, \\ S_3 &= 4U_{i-1,j+1} + 4U_{i,j+2} + U_{i-1,j} + U_{i-1,j+2} + U_{i+1,j+2}, \\ S_4 &= 4U_{i+2,j+1} + 4U_{i+1,j+2} + U_{i+2,j} + U_{i,j+2} + U_{i+2,j+2}. \end{aligned}$$

By determining the inverse of coefficient matrix in equation (42), the general scheme of this iterative method can be rewritten as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix} = \frac{1}{2079} \begin{bmatrix} 699 + S_a \\ 99S_2 + S_b \\ 99S_3 + S_b \\ 99S_4 + S_a \end{bmatrix} \tag{43}$$

where

$$S_a = 28(S_2 + S_3) + 17(S_1 + S_4),$$
$$S_b = 28(S_1 + S_4) + 17(S_2 + S_3).$$

Consequently, the SOR iterative scheme for equation (43) can be written as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix}^{(k+1)} = \frac{\omega}{2079} \begin{bmatrix} 99S_1 + S_a \\ 99S_2 + S_b \\ 99S_3 + S_b \\ 99S_4 + S_a \end{bmatrix} + (1-\omega) \begin{bmatrix} U_{i,j} \\ U_{i+1,j} \\ U_{i,j+1} \\ U_{i+1,j+1} \end{bmatrix}^{(k)} \tag{44}$$

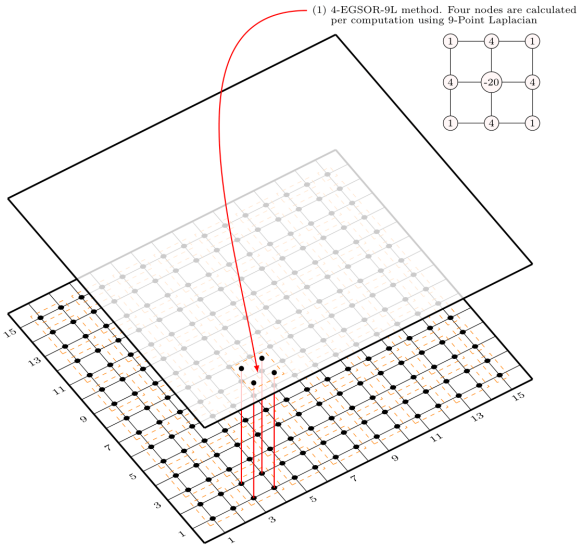


Figure 9 With 4-EGSOR-9L method, groups of four points are calculated using 9L approximation

4.7 The 4-EDGSOR-9L method

Let a group of four points with decoupled pairs as shown in Figure 10 be considered to form a system of linear algebraic equations and be given as [46]

$$\begin{bmatrix} 20 & -4 & 0 & 0 \\ -4 & 20 & 0 & 0 \\ 0 & 0 & 20 & -4 \\ 0 & 0 & -4 & 20 \end{bmatrix} \begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \\ U_{i+1,j} \\ U_{i,j+1} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \end{bmatrix} \tag{45}$$

where

$$S_1 = 4U_{i-1,j-1} + 4U_{i-1,j+1} + 4U_{i+1,j-1} + U_{i-1,j} + U_{i,j-1},$$
$$S_2 = 4U_{i,j+2} + 4U_{i+2,j+2} + 4U_{i+2,j} + U_{i+2,j+1} + U_{i+1,j+2},$$

and

$$S_3 = 4U_{i,j-1} + 4U_{i+2,j-1} + 4U_{i+2,j+1} + U_{i+2,j} + U_{i+1,j-1},$$
$$S_4 = 4U_{i-1,j} + 4U_{i-1,j+2} + 4U_{i+1,j+2} + U_{i-1,j+1} + U_{i,j+2}.$$

By splitting equation (45), this leads to a decoupled group (2 x 2) that can be written as

$$\begin{bmatrix} 20 & -4 \\ -4 & 20 \end{bmatrix} \begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \tag{46}$$

and

$$\begin{bmatrix} 20 & -4 \\ -4 & 20 \end{bmatrix} \begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix} = \begin{bmatrix} S_3 \\ S_4 \end{bmatrix}. \tag{47}$$

Then, determine the inverse of coefficient matrix in equations (46) and (47) toobtain

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix} = \frac{1}{96} \begin{bmatrix} 5 & 1 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} S_1 \\ S_2 \end{bmatrix} \tag{48}$$

and

$$\begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix} = \frac{1}{96} \begin{bmatrix} 5 & 1 \\ 1 & 5 \end{bmatrix} \begin{bmatrix} S_3 \\ S_4 \end{bmatrix}, \tag{49}$$

respectively. The SOR iterative scheme for equation (48) and (49) can be written independently as

$$\begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix}^{(k+1)} = \frac{\omega}{96} \begin{bmatrix} 5S_1 + S_2 \\ S_1 + 5S_2 \end{bmatrix} + (1-\omega) \begin{bmatrix} U_{i,j} \\ U_{i+1,j+1} \end{bmatrix}^{(k)} \tag{50}$$

and

$$\begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix}^{(k+1)} = \frac{\omega}{96} \begin{bmatrix} 5S_3 + S_4 \\ S_3 + 5S_4 \end{bmatrix} + (1-\omega) \begin{bmatrix} U_{i+1,j} \\ U_{i,j+1} \end{bmatrix}^{(k)}. \tag{51}$$

The configuration spaces are divided into two types of nodes, i.e. white and black nodes, where the solutions of any group of nodes are based on decoupled pairs. The 4-EDGSOR-9L may utilize either equation (50) or (51). For groups near to the boundaries, with only two or single nodes, direct method is applied.

5.0 SIMULATION RESULTS AND DISCUSSION

The simulations of robot path planning are run in a static environment using two simple configuration spaces i.e. Case 1 and Case 2. The considered methods are examined against several sizes of configuration space i.e. 300 by 300, 600 by 600, 900 by900, 1200 by 1200, 1500 by 1500 and 1800 by 1800. The configuration space consists of a goal point, obstacles, inner walls and outer boundary walls. In the initial setup, the obstacles, inner and outer walls are fixed with high potential values, whilst the goal point is set to a fixed lowest potential value, and no initial values are assigned to all other free spaces. The computation are carried out using Windows XP machine running on Intel Core 2 Duo CPU at 2.5GHz speed equipped with 1GB of RAM. The codes are written in Pascal, and the generation of paths are simulated in the self-developed software namely Robot 2D Simulator.

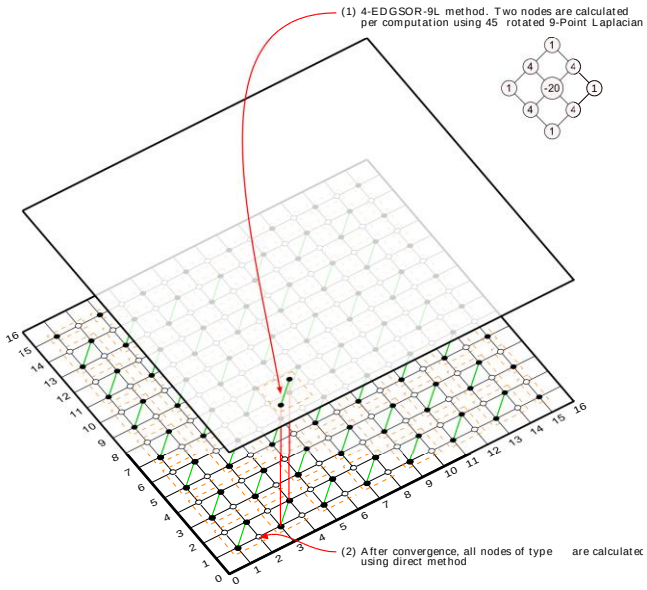


Figure 10 With 4-EDGSOR-9L method, groups of four points consists of decoupled pairs

The numerical representation used for these experiments is important. The iteration technique is terminated when there is no change of potential values from one iteration process to the next. A very high precision is required, thus the implementation used an 8 bytes variable storage of type Double for storing each potential value. The range value for Double is5.0 x 10^{-324} to 1.7 x 10^{-308}, and it can store up to 15 significant digits. The convergence criterion is set to a very small error tolerance i.e. 1.0^{-15}, since lower precision is not sufficient to avoid flat areas in the resulting potential values.

For performance comparisons, the HPF are calculated using the considered iterative methods as described in Section 4.3 to 4.7 (i.e. FSSOR, HSSOR, 4-EGSOR, 4-EDGSOR, FSSOR-9L, HSSOR-9L, 4-EGSOR-9L and the newly suggested 4-EDGSOR-9L). Once the HPF in the configuration space are established, the paths can be generated by following the gradient using the Gradient Descent Search (GDS) from the start point to the specified goal point. In a static situation where the goal point and obstacles are fixed, the solutions may be computed and then reused as often as desired. Unless the obstacles or goal point change position, the solution need not be recomputed. The position of start point does not affect the computation of the solutions.

The path planning simulations are conducted in several runs. Each run successfully generates smooth path from the start point to the goal point. In the generated paths shown in Figures 11 and 12, the solid square in green colour denotes start point, whilst the solid circle in red colour denotes goal point.

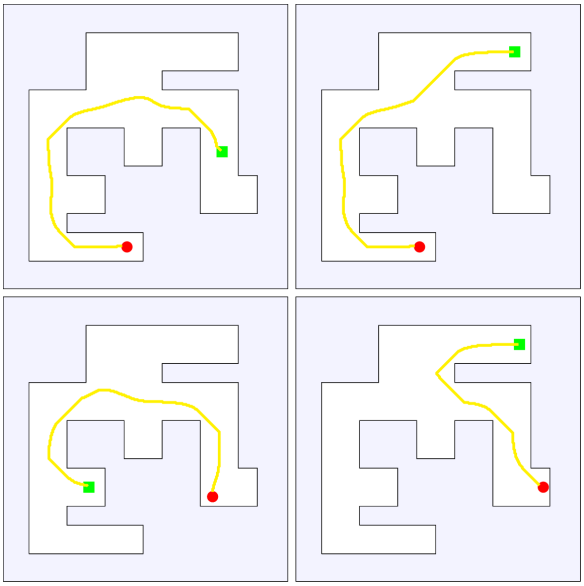


Figure 11 The generated paths for Case 1

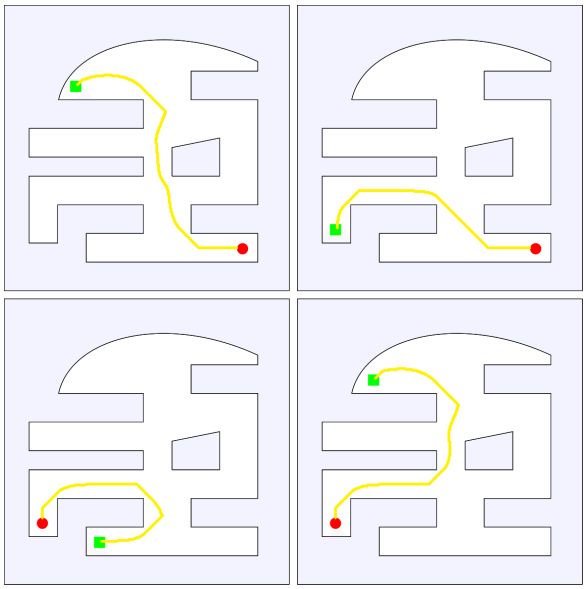


Figure 12 The generated paths for Case 2

Based on the simulation results, the performance in terms of number of iterations and CPU time for the considered methods are tabulated in Tables 1 and 2, respectively. Whilst, Table 3 shows the reduction percentages in terms of number of iterations and CPU time between the currently suggested method and the previous methods. As shown in Tables 1 and 2, the newly suggested 4-EGSOR-9L method is clearly faster than the previous methods. Also, 4-EDGSOR-9L method reduced the number of iterations significantly. Overall, the block variants of half-sweep iterative methods (the 4-EDGSOR and 4-EDGSOR-9L) give the best performance. The 4-EDGSOR-9L method that based on 9L approximation, however, is slightly faster than the 4-EDGSOR method. Compared to the standard FSSOR, the suggested 4-EDGSOR-9L method gives the best complexity reduction percentages by reducing the iteration number and execution time approximately by 65.0% to 66.9% and 70.8% to 76.5%, respectively.

Table 1 Number of iterations

		N					
	Methods	**300**	**600**	**900**	**1200**	**1500**	**1800**
CASE 1	**FSSOR**	2804	10686	23218	40322	61840	87686
	HSSOR	1426	5473	11915	20698	31780	45065
	EGSOR	1428	5533	11979	20880	31927	45385
	EDGSOR	1069	4175	9070	15810	24228	34423
	FSSOR-9L	2349	8964	19485	33837	51905	73601
	HSSOR-9L	1183	4584	9998	17375	26674	37826
	EGSOR-9L	1894	7263	15770	27426	42042	59681
	EDGSOR-9L	949	3697	8094	14039	21601	30616
CASE 2	**FSSOR**	2189	8331	18130	31473	48258	68431
	HSSOR	1109	4260	9299	16153	24786	35158
	EGSOR	1105	4327	9335	16313	24884	35427
	EDGSOR	822	3250	7068	12338	18876	26845
	FSSOR-9L	1834	6985	15214	26409	40501	57441
	HSSOR-9L	914	3562	7793	13547	20793	29507
	EGSOR-9L	1475	5657	12313	21399	32802	46556
	EDGSOR-9L	734	2872	6317	10951	16850	23880

Table 2 CPU times (in seconds)

		N					
	Methods	**300**	**600**	**900**	**1200**	**1500**	**1800**
CASE 1	**FSSOR**	5.64	99.78	553.55	1720.28	4190.00	8605.36
	HSSOR	1.52	25.44	147.75	471.20	1137.26	2328.05
	EGSOR	3.20	49.80	270.48	841.89	2030.00	4179.11
	EDGSOR	1.53	23.94	137.70	432.63	1047.69	2134.50
	FSSOR-9L	5.94	88.59	494.72	1533.14	3707.52	7621.81
	HSSOR-9L	1.33	24.53	146.52	465.52	1124.09	2309.01
	EGSOR-9L	5.92	92.20	493.67	1533.67	3695.33	7561.14
	EDGSOR-9L	1.55	24.58	141.59	439.11	1066.86	2172.61
CASE 2	**FSSOR**	4.59	79.06	446.23	1398.06	3415.03	7017.78
	HSSOR	1.19	21.52	121.83	383.69	927.31	1902.27
	EGSOR	2.53	40.70	216.81	677.38	1632.20	3371.28
	EDGSOR	1.22	19.48	110.59	346.75	838.91	1712.59
	FSSOR-9L	4.86	73.03	404.78	1249.61	3028.86	6226.06
	HSSOR-9L	1.11	20.09	120.94	380.91	926.39	1883.69
	EGSOR-9L	4.72	74.00	395.02	1223.61	2948.91	6038.83
	EDGSOR-9L	1.23	19.84	113.48	351.05	851.61	1737.94

Table 3 Reduction percentages

	Number of iterations	CPU time (in seconds)
Methods	**%**	**%**
FSSOR	0	0
HSSOR	48.6 – 49.3	72.5 – 74.5
EGSOR	48.0 – 49.8	40.9 – 53.5
EDGSOR	60.8 – 62.5	71.2 – 76.6
FSSOR-9L	16.0 – 16.2	(11.0) – 13.8
HSSOR-9L	56.8 – 58.3	71.5 – 78.8
EGSOR-9L	31.9 – 32.6	(12.4) – 18.5
EDGSOR-9L	65.0 – 66.9	70.8 – 76.5

6.0 CONCLUSION

In conclusion, the newly suggested 4-EDGSOR-9L method manages to speed up the computation of HPF significantly. The application of GDS to the obtained HPF quickly generates smooth paths from start point to the goal point in both Case 1 and Case 2. In computing the HPF, compared to the standard point iterative method that employsfull-sweep iteration, the half-sweep approach reduced the computational complexity by approximately 50%.In comparison to the point iterative methods, their corresponding block variants further reduce the computational complexity. Thus, they greatly speed up the convergence of the iterations process to obtain the HPF, thus consequently further improve the overall performance of the path planning algorithm. Furthermore, although the computational complexity of the iterative methods based on 9Lare slightly higher than the 5L methods, the 9L methods converge slightly faster, since they obtain more accurate solutions than their corresponding 5L methods due to greater number of molecules involved per node calculation. Nevertheless, the performance differences between the 5L methods and their corresponding9L methods are minimal. In the future, faster iterative methods that employ quarter-sweep approach [47] and its block variants[48] would be recommended.

Acknowledgement

We are grateful for the Universiti Malaysia Sabah financial support to this study.

References

[1] Connolly, C. I., Burns, J., and Weiss, R. 1990. Path planning using Laplace's equation. *Proceedings of the IEEE International Conference on Robotics and Automation, May 13-18, 1990, Cincinnati, USA*. 2102-2106.

[2] Akishita, S., Kawamura, S., and Hayashi, K. 1990. Laplace Potential For Moving Obstacleavoidance And Approach Of A Mobile Robot. *Japan-USA Symposium on FlexibleAutomation, July 9-13, 1990, Kyoto, Japan*. 139-142.

[3] Barraqu and, J., Langlois, B., and Latombe, J. C. 1992. Numerical Potential Field Techniques For Robot Path Planning. *IEEE Transactions on Systems, Man and Cybernetics*. 22(2): 224-241.

[4] Sasaki, S. 1998. A Practical Computational Technique For Mobile Robot Navigation. *Proceedings of the IEEE International Conference on Control Applications, Sep 4, 1998, Trieste, Italy*. 2: 1323-1327.

[5] Saudi, A. and Sulaiman, J. 2009. Efficient Weighted Block Iterative Method for Robot Path Planning Using Harmonic Functions. *Proceedings of the 2nd International Conference on Control, Instrumentation and Mechatronic Engineering (CIM09), Malacca, Malaysia.* June 2-3, 2009.

[6] Saudi, A. and Sulaiman, J. 2009. Path Planning for Mobile Robot with Half-Sweep Successive Over-Relaxation (HSSOR) Iterative Method. *Proc. of the Symposium on Progress in Information & Communication Technology, Kuala Lumpur (SPICT'09)*. 57-62.

[7] Saudi, A. and Sulaiman, J. 2010. Robot Path Planning based on Four Point-EGSOR Iterative Method. *Proc. of the 2010 IEEE Conference on Robotics Automation and Mechatronics (RAM)*. 476 – 481. ISBN: 978-1-4244-6503-3.

[8] Saudi, A. and Sulaiman, J. 2012. Robot Path Planning via EGSOR Iterative Method Using Nine-Point Laplacian. *Proc. of the 3rd Int. Conf. on Intelligent Systems, Modeling and Simulation (ISMS2012), Kota Kinabalu, Malaysia, 8-10 Feb 2012*. 61 – 66. ISBN: 978-1-4673-0886-1.

[9] Saudi, A. & Sulaiman, J. 2012. Path Planning for Mobile Robot using 4EGSOR via Nine-Point Laplacian (4EGSOR9L) Iterative Method. *International Journal of Computer Applications.* 53(16): 38-42. ISBN: 973-93-80870-44-0.

[10] Saudi, A. & Sulaiman, J. 2012. Path Planning for Indoor Mobile Robot using Half-Sweep SOR via Nine-Point Laplacian (HSSOR9L). *IOSR Journal of Mathematics (IOSR-JM), ISSN: 2278-5728*. 3(2) Sep-Oct. 2012: 1-7. ISSN: 2278-5728.

[11] Connolly, C. I. and Grupen, R. A. 1993. The Applications Of Harmonic Functions To Robotics. *Journal of Robotic Systems.* 10(7): 931-946.

[12] Zelek, J. S. and Levine, M. D. 2000. Local-Global Concurrent Path Planning And Execution. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans.* 30(6): 865-870.

[13] Khatib, O. 1985. Real-Time Obstacle Avoidance For Manipulators And Mobile Robots. *Proceedings of the IEEE International Conference on Robotics and Automation, Mar 25-28, 1985, St. Louis, USA*. 2: 500-505.

[14] Koditschek, D. E. 1987. Exact Robot Navigation By Means Of Potential Functions: Some Topological Considerations. *Proceedings of the IEEE International Conference on Robotics and Automation, March 31 - April 3, 1987, Raleigh, USA*. 4: 1-6.

[15] Rimon, E. and Koditschek, D. E. 1992. Exact Robot Navigation Using Artificial Potential Functions. IEEE Transactions on Robotics and Automation. 8(5): 501-518.

[16] Kim, J. O. and Khosla, P. K. 1992. Real-Time Obstacle Avoidance Using Harmonic Potential Functions. *IEEE Transactions on Robotics and Automation.* 8(3): 338-349.

[17] Waydo, S. and Murray, R. M. 2003. Vehicle Motion Planning Using Stream Functions. *Proceedings of the IEEE International Conference on Robotics and Automation, Sep 14-19, 2003, Taipei, Taiwan*. 2: 2484-2491.

[18] Rosell, J. and Iniguez, P. 2005. Path Planning Using Harmonic Functions And Probabilistic Cell Decomposition. *Proceedings of the IEEE International Conference on Robotics and Automation, April 18-22, 2005, Barcelona, Spain.* 1803-1808.

[19] Daily, R. and Bevly, D. M. 2008. Harmonic Potential Field Path Planning For High Speed Vehicles. *Proceedings of the IEEE American Control Conference, June 11-13, 2008, Seattle, USA*. 4609-4614.

[20] Garrido, S., Moreno, L., Blanco, D., and Martin Monar, F. 2010. Robotic Motion Using Harmonic Functions And Finite Elements. *Journal of Intelligent and Robotic Systems*. 59(1): 57-73.

[21] Szulczynski, P., Pazderski, D., and Kozlowski, K. 2011. Real-Time Obstacle Avoidance Using Harmonic Potential Functions. *Journal of Automation Mobile Robotics and Intelligent Systems.* 5: 59-66.
[22] Yang, F. and Ariyur, K. B. 2011. Laplacian Path Planning: Implementation And Generalizations. *Proceedings of the Infotech@Aerospace 2011 Conference, March 29-31, 2011, St. Louis, MO, USA*. 1-9.
[23] Pedersen, M. D. and Fossen, T. I. 2012. Marine Vessel Path Planning And Guidance Using Potential Flow. *Proceedings of the 9th IFAC Conference on Manoeuvring and Control of Marine Craft, Sep 19-21, 2012, Arenzano, Italy*. 188-193.
[24] Liang, X., Wang, H., Li, D., and Liu, C. 2014. Three-Dimensional Path Planning For Unmanned Aerial Vehicles Based On Fluid Flow. *Proceedings of the IEEE Aerospace Conference, March 1-8, 2014, Big Sky, USA*. 1-13.
[25] Kress, R. 1998. Numerical Analysis. *New York: Springer-Verlag*.
[26] Saad, Y. and Van Der Vorst, H. A. 2000. Iterative Solution Of Linear Systems In The 20th Century. *Journal of Computational and Applied Mathematics.* 123(1-2): 1-33.
[27] Abdullah, A. R. 1991. The Four Point Explicit Decoupled Group (EDG) Method: A Fast Poisson Solver. *International Journal of Computer Mathematics.* 38(1-2): 61-70.
[28] Evans, D. J. 1985. Group Explicit Iterative Methods For Solving Large Linear Systems. *International Journal of Computer Mathematics.* 17(1): 81-108.
[29] Yousif, W. S. and Evans, D. J. 1995. Explicit De-coupled Group Iterative Methods And Their Parallel Implementations. *Parallel Algorithms and Applications*. 7(1-2): 53-71.
[30] Sulaiman, J., Othman, M., and Hasan, M. K. 2009. Nine Point-EDGSOR Iterativemethodfor The Finite Element Solution Of 2D Poisson Equations. O. Gervasi, D. Taniar,B. Murgante, A. Lagana, Y. Mun, and M. L. Gavrilova. (eds.). Lecture Notes in Computer Science 5592: Computational Science and Its Applications - ICCSA 2009. 764-774. Berlin *Heidelberg: Springer-Verlag.*
[31] Akhir, M. K. M., Othman, M., Abdul Majid, Z., Suleiman, M., and Sulaiman, J. 2012. Four Point Explicit Decoupled Group Iterative Method Applied To Two-Dimensional Helmholtz Equation. *International Journal of Mathematical Analysis.* 6(20): 963-974.
[32] Akhir, M. K. M., Othman, M., Sulaiman, J., Abdul Majid, Z., and Suleiman, M. 2011. Half-Sweep Modified Successive Overrelaxation For Solving Two-Dimensional Helmholtz Equations. *Australian Journal of Basic and Applied Sciences*. 5(12): 3033-3039.
[33] Fauzi, N. I. M. and Sulaiman, J. 2012. Half-sweep Modified Successive Overrelaxtion Method For Solving Second Order Two-Point Boundary Value Problems Using Cubic Spline. *International Journal of Contemporary Mathematical Sciences.* 7(32): 1579-1589.
[34] Muthuvalu, M. S. and Sulaiman, J. 2012. Half-Sweep Geometric Mean Iterative Method For The Repeated Simpson Solution Of Second Kind Linear Fredholm Integral Equations. *Proyecciones.* 31(1): 65-79.
[35] Sulaiman, J., Hasan, M. K., Othman, M., and Karim, S. A. A. 2014. Implicit Finite Difference Solutions Of One-Dimensional Burgers' Equation Using Newton-HSSOR Method. A. Kilicman, W. J. Leong, and Z. Eshkuvatov. (eds.). *International Conference on Mathematical Sciences and Statistics 2013.* 285-295. Singapore: Springer-Verlag.
[36] Karnik, M., Dasgupta, B., and Eswaran, V. 2002. A Comparative Study Of Dirichlet And Neumann Conditions For Path Planning Through Harmonic Functions. In P. M. A. Sloot, A. G. Hoekstra, C. J. K. Tan, and J. J. Dongarra. (eds.). Lecture Notes in Computer Science 2330: Computational Science - ICCS 2002. 442-451. *Berlin Heidelberg: Springer-Verlag.*
[37] Le Veque, R. J. 2007. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-state and Time-dependent Problems*. Philadelphia: Society for Industrial Applied Mathematics.
[38] Zachmanoglou, E. C. and Thoe, D. W. 2012. Introduction to Partial Differential Equations with Applications. New York: Dover Publications.
[39] Souccar, K., Coelho, J., Connolly, C., and Grupen, R. 1998. Harmonic Functions For Path Planning And Control. Practical Motion Planning In Robotics: Current Approaches And Future Directions. 277-302. *Chichester: John Wiley*.
[40] Michel, O. 2004. WebotsTM: Professional Mobile Robot Simulation. *International Journal of Advanced Robotics Systems.* 1(1): 39-42.
[41] Fletcher, C. A. J. 1984. Computational Galerkin Methods. Springer Series in Computational Physics. *New York: Springer-Verlag*.
[42] Sulaiman, J., Othman, M., and Hasan, M. K. 2007. Red-Black Half-Sweep Iterative Method Using Triangle Finite Element Approximation For 2D Poisson Equations. Y. Shi, G. D. van Albada, J. Dongarra, and P. M. A. Sloot. (eds.). Lecture Notes in Computer Science 4487: Computational Science - ICCS 2007. 326-333. *Berlin Heidelberg: Springer-Verlag*.
[43] Young, D. M. 1950. Iterative *Methods for Solving Partial Difference Equations of Elliptic Type*. PhD Thesis. Harvard University.
[44] Ali, N. H. M. and Abdullah, A. R. 1998. New Parallel Nine-Point Poisson Solver. *Journal of Information Technology*. 2(10): 1-9.
[45] Ling, S. T. and Ali, N. H. M. 2013. New High Order Group Iterative Schemes In The Solution Of Poisson Equation. *International Journal of Mathematical, Computational, Physical and Quantum Engineering*. 7(12): 1170-1175.
[46] Ali, N. H. M. 2002. Reka Bentuk Algoritma Blok Baru Stensil 9-Titik Dalam Masalah Nilai Sempadan. *Matematika.* 18(1): 45-62.
[47] Fauzi, N. I. M. and Sulaiman, J. 2013. Quarter-Sweep Gauss-Seidel Method With Quadratic Spline Scheme Applied To Fourth Order Two-Point Boundary Value Problems. *Proceedings of the 20th National Symposium on Mathematical Sciences: Research in Mathematical Sciences: A Catalyst for Creativity and Innovation, Dec 18-20, 2013, Putrajaya, Malaysia.* 1522: 735-743.
[48] Sulaiman, J., Othman, M., and Hasan, M. K. 2010. MEGSOR Iterative Scheme For The Solution Of 2D Elliptic PDE's. *World Academy of Science, Engineering and Technology.* 38: 418-424.