Numerical simulations of agent navigation via Half-Sweep Modified Two-Parameter Over-Relaxation (HSMTOR)

F A Musli 1, J Sulaiman 2 and A Saudi 1
1Faculty of Computing and Informatics, Universiti Malaysia Sabah, Kota Kinabalu, Malaysia.
2Faculty of Science and Natural Resources, Universiti Malaysia Sabah, Kota Kinabalu, Malaysia.

E-mail: 1farhahathirahmusli@ymail.com, 2jumat@ums.edu.my, 3azali@ums.edu.my

Abstract. The research on the efficiency of route navigation has been continuously developing. Especially, the capability of the generated route to provide a collision-free route for an agent to move in a particular environment. Thus, this study attempts to solve the route navigational problem iteratively via a numerical method. A new method called Half-Sweep Modified Two-Parameter Over-Relaxation (HSMTOR) is used to solve the navigational problems. For numerical simulation purposes, HSMTOR is used to obtain Laplace's equation solutions called harmonic functions. A gradient descent search algorithm then utilizes the harmonic functions to provide a smooth and collision-free route for an agent to commute inside the environment. In addition, the formulation of the HSMTOR iterative method is presented. Several numerical experiments and simulations are conducted in order to verify the efficiency of the proposed method. The result shows that the proposed method performed better than the existing methods such as full-, half-sweep for Modified Successive Over-Relaxation, Modified Accelerated Over-Relaxation and Modified Two-Parameter Over-Relaxation respectively (FSMSOR, HSMOSR, FSMAOR, HSMAOR and FSMTOR).

1. Introduction
Navigational route problems play a significant role in automated applications, designed animations etc. [1]. Therefore, the research activities in this field have been relentlessly expanding over the years. The ultimate target of agent navigation is to carry out the mission of finding a route safely without any collision in the desired environment. Inevitably, many algorithms aim to solve this navigational route problems. However, several fail to generate the route due to limitations, such as when the configuration space's size grows exponentially. This situation will cause the difficulty and complexity of the route navigation. There are two types of route planners called global and local methods. A global method is known as the hassle-free planner as the agent has the environment's information beforehand. However, the global method taxing too much of computational time and expensive [2]. Meanwhile, the local method is the opposite of the global method, where it allows an agent to move freely in a dynamic environment. Nevertheless, the local method faces some difficulties when the targeted destination is far away and complex environment. This situation may cause the route exploration to be failed [3]. Thus, this study is conducted to solve the route navigational problem by employing the global route generation method in designed environments. An HSMTOR tested in the navigational problem to generate the optimum route in several sizes of environments with various setup of obstacles. Furthermore, the efficiency of the method is compared with the previous method.
2. Literature reviews
In the literature, the solution for Laplace's equation is well-known as harmonic functions, which are stated to have several properties that are quite helpful in robotics and automated applications [4]. Prior, solving navigational problems via the implementation of the harmonic function's solution has been executed by Connolly and Akishita [5]. Numerical methods are the standard method used for computing the harmonic functions compare to the analytical methods. Numerical methods are known as methods with fast computational capability and very efficient in solving boundary problems [6]. Jacobi, Gauss-Seidel (GS), and Successive Over Relaxation (SOR) were the standard methods to solve navigational route problems. Based on the previous study, the performance of SOR is remarkably faster than the other two standard methods [7]. Later, a family member of a numerical method called the finite element method is used to solve Laplace's equation [8]. Despite several attempts at solving harmonic functions via analytical method [9]. However, the problem arises when it comes to arbitrarily shaped obstacles. Afterward, harmonic functions are applied for the navigational related problem, such as the real-time obstacles avoidance mechanism [10-11]. Recently, a new family member of Over-Relaxation called Two-Parameter Over Relaxation (TOR) has been introduced [12]. TOR has been proved as an effective method to solve Partial Differential Equations (PDEs) [13]. Thus, in this study, we tried to implement the modified version of this method in solving route navigational problems. As follows, in this study, a new method called HSMTOR is proposed to compute the solution of Laplace's equation and route navigation at once.

3. Research Methodology
3.1 Route Navigational Planning
The main concern of route navigation is to provide a collision-free route for an agent to move from a starting point towards the targeted destination within a particular workspace. With the aid of mathematical reasoning of Laplace's solutions, a route generation is obtained and useful for the agent to navigate. The idea of how the route navigation work is based on the analogy of heat distributions discussed in section 3.3.

3.2 Harmonic Functions
Laplace equation is known as an elliptic equation under the family of PDEs [14]. A continuous function satisfying Laplace's equation in an open region is called a harmonic function [15]. Harmonic functions stated to have essential properties in solving route navigational problems. One of its advantages is that it allows the agent exempt from experiencing the local minima problems [16]. A harmonic function on a domain is a function that satisfies Laplace's equation

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

(1)

where \( x_i \) and \( n \) equivalently denote the i-th Cartesian coordinate and the dimension of the environment. The boundary incorporates the inner and outer wall for the route exploration inside the workspace, including some obstacles. Thus, with Laplace's equation's implementation as a constraint, any local-minimum formation within the workspace shall be avoided.

3.3 Heat Transfer Analogy
The idea of route navigational problem in the described environment is based on the analogy of the steady-state heat-transfer problem. Steady-state heat transfer or heat distribution is the best to describe how route navigation works. The outer, inner boundary walls and all obstacles inside the virtual environment are treated as heat sources, whilst the goal point will act as a heat sink that draws the heat in. This process eventually enforces a temperature distribution where the heat shall move from higher heat sources to lower where the heat sink.

In the mathematical description, the harmonic potentials distribution in the workspace is derived from the solutions of Laplace’s equation. Through an iterative process, harmonic potentials are
computed until the specified convergence criterion is satisfied. Based on the analogy given, with the aid of a gradient descend search algorithm, the agent shall be able to trace the harmonic potential functions distribution where it flows from higher (start point) to lower value (goal point) of the solutions. Thus, it allows the agent to navigate inside a particular environment without any collision.

3.4 Formulation of HSMTOR Iterative Method
This study presented a faster iterative solver to compute Laplace's equation solutions using the MTOR iterative method. MTOR is known as the generalization of the TOR method. TOR has been proved as an iterative method that converges faster than Jacobi, GS, SOR, and AOR [17]. When it comes to a modified (red and black strategy) algorithm, we believe it will give the best performance as its origin. Thus, this study is conducted to justify the efficiency of the proposed method. The formulation of the MTOR iterative method is identical to the TOR method except for the weighted parameters and the red and black strategy implementation. The red and black strategy is likely known as an approach with spectacular parallel computing and convergence rate capability [18]. The analogy of the red-black strategy can be described as a checkerboard game. Each red square has its black squares surrounding it, whether from north, south, east, and west and vice versa. Meanwhile, in numerical executions, the process depends on whether the node is odd or even. The computation process will start with a red node (even) first at the bottom left row until all red nodes are updated. Simultaneously, the black nodes (odd) shall go through the same process as the red nodes. This process allows alternating phases and repeat until it converges, as depicted in figure 1. Whilst, figure 2 shows the molecule nodes for both red and black computations.

![Figure 1. The computational grid for modified (red and black strategy) iteration](image1.png)

![Figure 2. The molecule nodes for (a) red computation (b) black computation, respectively](image2.png)
A MTOR method is employed in traditional full and half-sweep iteration using a five-point discretization scheme. All inner nodes in the grid will be computed during the full-sweep iteration process, whilst the half-sweep iteration will compute only half of the total nodes [19]. Figures 3 and 4 illustrate the computational node and stencil point of both full and half sweep iteration methods. Meanwhile, figures 5 and 6 represent the molecule or stencil point of full- and half-sweep iteration.

![Computational Grids](image)

**Figure 3.** The computational grid of full-sweep iteration  
**Figure 4.** The computational grid of half-sweep iteration

![Stencil Points](image)

**Figure 5.** The stencil point of (a) full-sweep and (b) full-sweep, respectively

The corresponding formulation of FSMTOR and HSMTOR iterative schemes are solved by the red and black strategy, as presented in equation (2-5), respectively.

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

in black nodes, whereas red nodes as
\[ U_{i,j}^{k+1} = \frac{r}{4} \left[ U_{i-1,j}^{(k+1)} - U_{i-1,j}^{(k)} \right] + \frac{s}{4} \left[ U_{i,j-1}^{(k+1)} - U_{i,j-1}^{(k)} \right] + \frac{\omega}{4} \left[ U_{i-1,j}^{(k)} + U_{i,j+1}^{(k)} + U_{i,j}^{(k)} \right] + (1 - \omega) U_{i,j}^{(k)} \] (4)

\[ U_{i,j}^{k+1} = \frac{r}{4} \left[ U_{i-1,j}^{(k+1)} - U_{i-1,j-1}^{(k)} \right] + \frac{s}{4} \left[ U_{i+1,j-1}^{(k+1)} - U_{i+1,j-1}^{(k)} \right] + \frac{\omega}{4} \left[ U_{i-1,j-1}^{(k)} + U_{i+1,j-1}^{(k)} + \ldots \right] \] (5)

where \( r, s, \omega \) and \( \omega' \) are denoted as the optimum relaxation parameters and defined in the range of \([0, 2)\). Note that if \( r = s \), the TOR method returns to the AOR method [20]. The iterations process is only terminated when there is no change of any node point from one iteration to the next.

3.5 The Simulation of Route Navigational

This study intends to plan a route for an agent to move within a particular workspace or environment. As for the simulation, the environment presented in 2D rectangular outer boundary walls consists of distinct inner walls and obstacles. Meanwhile, there are two points allowed inside the environment: green and red. Within the environment, the green point denotes the agent navigator (start point), whereas the red point denotes the goal or desired destination. Additionally, two static maps with area of 330x300, 660x600, 990x900, and 1320x1200 were chosen for simulation purposes [21]. The route navigational algorithm begins by finding the solution of Laplace’s equation using the previously discussed iterative methods with multiple weighted parameters. The process of iteration is repeated until the convergence condition is fulfilled. The resulting route is then traced using a gradient descent search. The implementation of the route navigation algorithm is described in Algorithm 1.

Algorithm 1: Route Navigation Algorithm

1) Load the selected map of the environment
2) Set the potential values of starting and goal point
3) Compute the Laplace’s equation to obtain harmonic potential function by using the considered methods until convergence criterion is satisfied.
4) Apply gradient descent searching to navigate the agent from start to goal point inside the environment
5) Display the generated route

4. Results

For the purpose of simulations, there are two different maps with several sizes used to test the methods’ efficiency. The maps consisted of start point, goal point, static obstacles and inner-outer boundary walls. The simulation with different start and target positions are tested, and the results show successful routes generation as shown in figure 6-7. The computation process was run on an Intel Core i5-3210M processor PC running at 3.1GHz speed with 8GB of RAM.
Table 1. Number of Iterations and Computational Time for Map 1.

| Map size | Methods  | Number of iterations | CPU time           |
|----------|----------|----------------------|--------------------|
| 330 x 300| FSMSOR   | 3832                 | 0 min, 1 sec, 434 ms |
|          | FSMAOR   | 3026                 | 0 min, 1 sec, 269 ms |
|          | FSMTOR   | 2827                 | 0 min, 1 sec, 223 ms |
|          | HSMSOR   | 1963                 | 0 min, 0 sec, 677 ms |
|          | HSMAOR   | 1526                 | 0 min, 0 sec, 581 ms |
|          | HSMTOR   | 1416                 | 0 min, 0 sec, 566 ms |
| 660 x 600| FSMSOR   | 14408                | 0 min, 21 sec, 309 ms |
|          | FSMAOR   | 11479                | 0 min, 18 sec, 937 ms |
|          | FSMTOR   | 10558                | 0 min, 18 sec, 133 ms |
|          | HSMSOR   | 7415                 | 0 min, 8 sec, 985 ms |
|          | HSMAOR   | 5885                 | 0 min, 7 sec, 802 ms |
|          | HSMTOR   | 5506                 | 0 min, 7 sec, 564 ms |
| 990 x 900| FSMSOR   | 31029                | 2 min, 13 sec, 808 ms |
|          | FSMAOR   | 25565                | 1 min, 54 sec, 120 ms |

Figure 6. The generated routes for Map 1

Figure 7. The generated routes for Map 2
The number of iterations and computational time (in seconds) required to compute numerical techniques is tabulated in tables 1 and 2. Note that the convergence criterion is set to a very small epsilon value where $\varepsilon = 1.0^{-15}$. Such that, very high precision is needed to avoid any rise of failure in finding route navigation. Tables 1 and 2 show that the HSMSOR iterative method is significantly superior to the previous method as it requires fewer iterations. Eventually, in terms of computational time, HSMTOR has the ability to iterate faster than the other methods. This reasonable reduction was able to achieve with the aid of a half-sweep scheme. As such, the proposed method HSMTOR is proved as an efficient method to solve agent navigational route problems.

### Table 2. Number of Iterations and Computational Time for Map 2.

| Map size  | Methods | Number of iterations | CPU time          |
|-----------|---------|----------------------|-------------------|
| 330 x 300 | FSMSOR  | 2974                 | 0 min, 1 sec, 44 ms |
|           | FSMAOR  | 2341                 | 0 min, 0 sec, 944 ms |
|           | FSMTOR  | 2182                 | 0 min, 0 sec, 923 ms |
|           | HSMSOR  | 1505                 | 0 min, 0 sec, 508 ms |
|           | HSMAOR  | 1167                 | 0 min, 0 sec, 451 ms |
|           | HSMTOR  | 1081                 | 0 min, 0 sec, 446 ms |
| 660 x 600 | FSMSOR  | 11178                | 0 min, 15 sec, 385 ms |
|           | FSMAOR  | 8888                 | 0 min, 13 sec, 804 ms |
|           | FSMTOR  | 8311                 | 0 min, 13 sec, 383 ms |
|           | HSMSOR  | 5744                 | 0 min, 6 sec, 830 ms |
|           | HSMAOR  | 4554                 | 0 min, 5 sec, 851 ms |
|           | HSMTOR  | 4254                 | 0 min, 5 sec, 686 ms |
| 990 x 900 | FSMSOR  | 24176                | 1 min, 45 sec, 365 ms |
|           | FSMAOR  | 19268                | 1 min, 28 sec, 790 ms |
|           | FSMTOR  | 18020                | 1 min, 23 sec, 204 ms |
|           | HSMSOR  | 12455                | 0 min, 45 sec, 127 ms |
|           | HSMAOR  | 9920                 | 0 min, 38 sec, 495 ms |
|           | HSMTOR  | 9274                 | 0 min, 35 sec, 883 ms |
| 1320 x 1200| FSMSOR  | 41388                | 5 min, 51 sec, 823 ms |
|           | FSMAOR  | 33007                | 5 min, 2 sec, 535 ms |
|           | FSMTOR  | 30906                | 5 min, 1 sec, 239 ms |
|           | HSMSOR  | 21348                | 2 min, 25 sec, 613 ms |
|           | HSMAOR  | 16998                | 2 min, 4 sec, 896 ms |
|           | HSMTOR  | 15912                | 1 min, 55 sec, 691 ms |
5. Conclusion and Future Work

Based on the previous section results, both full- and half-sweep MTOR are quite promising in solving the route navigational problems. This is due to the advanced techniques finding as well as the accessibility of powerful and faster machine these days. As for future work, further investigation on quarter-sweep iteration should be considered in order to speed up the convergence rate of the method.

References

[1] Ouarda H. 2010. *Int. J. Math. Model. Methods Appl. Sci.* **4**(3), 177–186
[2] Dorf R. C. 2018. The Electrical Engineering Handbook-Six Volume Set. (United States: CRC press).
[3] Nguyen H T and Le H X. 2016. Path planning and Obstacle avoidance approaches for Mobile robot. *IJCSI*. **13** (Preprint arXiv:1609.01935)
[4] Connolly C I, Burns J B and Weiss R. 1990. *Proc. IEEE Int. Conf. Robotic Autom.* 2102-2106
[5] Musli F A and Saudi A. 2019. *IEEE 9th Symp. Comp. Appl. & Ind. Elec* (ISCAIE) 335-339.
[6] Szukiewicz M K. 2017. *Brazilian J. of Chem. Eng.* **34**(3), 873-883.
[7] Dahalan A A, Saudi A, Sulaiman J and Din WR. J. 2017. *J. Phys. Conf. Ser.* 890.
[8] Garrido S, Moreno L, Blanco D and Monar FM. 2010. *J. of Intelligent and Robotic Syst.* **59**(1). 57-73.
[9] Daily R and Bevly D M. 2008. *American Control Conf.* 24609-4614
[10] Szulczyński P, Pazderski D and Kozlowski K. 2011. *J. Autom. Mob. Robot. Intell. Syst.* **5**. 59–66
[11] Yang F and Ariyur K. 2011. Proc. of the Infotech @ Aerospace 2011 Conf. 1–9.
[12] Kuang J X. 1983. *J. of Shanghai Teachers College (Nat. Sci. Ed.)*. **4**. 1-1.
[13] Li W and You Z Y. 1998. *J. of Comp. Math.* **16**(4). 367-374.
[14] Fulton S R, Fokas A S and Xenophontos C A. 2004. *J. of Comp. and Appl. Math.* **167**(2), 465-483.
[15] Helms L L. 2009. *Potential theory*. (Springer: London)
[16] Suparmin S and Saudi A. 2017. 2nd Int. Conf. on Automatic Control and Intelligent Sys. 46-51
[17] Zhang L T and Gu T X. 2017. *Math. Comput. Appl.* **22**(1), 20.
[18] Li R, Gong L and Xu M. 2020. *J.of Supercomputing*. **76**(12). 9585-9608
[19] Akhir M K M, Othman, M , Sulaiman J , Majid Z A, and Suleiman M. 2012. *J. of App. Math. and Stat*. 29, 101-109
[20] Kuang J and Ji J. 1988. *J. of Comput. Appl. Math.* **24**(1-2), 3-12.
[21] Silva Jr E P, Engel PM, Trevisan M and Idiart M A. 2002. *Robotics and Autonomous Sys.* **40**(1). 25-42.