Laplacian Behaviour-Based Control for Robot Path Planning using Full-Sweep Successive Over-Relaxation via Nine-Point Laplacian (FSSOR9L)

Azali Saudi

Jumat Sulaiman

School of Science and Technology
Universiti Malaysia Sabah
88400 Kota Kinabalu, Sabah
Malaysia

Abstract

This paper proposed a behaviour-based paradigm approach to the path planning problem of a mobile robot utilizing fast iteration technique to compute the configuration space rapidly. The technique employs Laplacian Behaviour-Based Control (LBBC) for robust exploration of the configuration space of the robot. The LBBC relies on Laplace's equation that constraint the potential function in the configuration space of the robot. The solution of Laplace's equation which represents the potential values in the configuration space is solved using Full-Sweep Successive Over-Relaxation via Nine-Point Laplacian (FSSOR9L) for fast computation. Several experiments were carried out to demonstrate the effectiveness of LBBC using fast FSSOR9L iterative method to compute the potential values rapidly and generate path for the robot successfully even in complex environment.

1. Introduction

In order to build a truly autonomous mobile robot, it must have the capability to efficiently and reliably plan a route from start to the goal point without colliding with obstacles in between. The effort of providing the robot with this path planning capability is considered one of the most challenging tasks in robotics field. Path planning algorithm attempts to deal with the problem of establishing a medium of communication between start and goal point, so that the robot can traverse the field safely. Various algorithms exist trying to solve this problem but all have shortcomings. The difficulty is due to the complexity of path planning problem, where it increases exponentially with the dimension of the configuration space.

In order to ensure completeness, every point in the configuration space has to be considered in the computation. Many global path planning methods presuppose a complete representation of the configuration space. Their main drawbacks, is that at best they are computationally expensive and often intractable. Potential field and bug approaches are local methods that do not make this assumption but are not complete methods. Thus, produce the occurrence of local minima or loops that will often cause this class of path planners to fail.

This work attempts to solve robot path planning problem by employing global method to generate path in complex environment. By applying Laplacian Behaviour-Based Control (LBBC), a robust searching algorithm will quickly generate path from starting to goal point configuration. Based on the theory of heat transfer, the environment is modeled as a configuration space, in which temperature distribution at each point will be used by the LBBC to guide its searching. The solutions of Laplace's equation, also known as harmonic functions, can be used to represent temperature values in the configuration space. For fast computation, the temperature distribution is numerically computed by solving Laplace's equation using weighted iterative method based on nine-point formula. In this work, several experiments were conducted to study the performance of using fast Full-Sweep Successive Over-Relaxation via Nine-Point Laplacian (FSSOR9L) iterative method for rapid computation of the solution of Laplace's equation, and to investigate the effectiveness of LBBC to generate path in several sizes of environment.

2. Literature Review

In [1], Connolly and Gruppen reported that harmonic functions have a number of properties useful in robotic applications. The use of potential functions for robot path planning, as introduced by Khatib [5], views every obstacle to be exerting a repelling force on an end effector, while the goal exerts an attractive force.

Koditschek [6], using geometrical arguments, showed that, at least in certain types of domains, there exists potential functions which can guide the effector from almost any point to a given point. These potential fields approach to path planning, however, suffer from the spontaneous creation of local minima. Connolly et al. [7] and Akishita et al. [8] independently developed a global method using solutions to Laplace's equations for path planning to generate a smooth, collision-free path. The potential field is computed in a global manner, i.e. over the entire region, and the harmonic solutions to Laplace's equation are used to find the path lines for a robot to move from the start point to the goal point. In previous work, [19] and [20], the Laplace's equation was solved numerically via block iterative method, in which the computation speed of the potential field was improved tremendously. This global method, however, suffer from the occurrence of flat region in complex environment which caused the path generation algorithm to fail. Several other methods are also proposed for solving path planning problem. In [12], an algorithm that employs distance transform method is reported. Jan et al. [13] conducted researches on utilizing geometry maze routing algorithm. The work by Bhattacharya and Gavrilova [14] uses Voronoi Diagram to solve path planning problem. In [15], genetic algorithm through evolutionary process was used for mobile robot path planning.

3. Laplacian Behaviour-Based Control

Traditional approach robot programming assumes the availability of a complete and accurate model of the robot and its environment, relying on planners to generate actions [16]. Unfortunately, this approach has several disadvantages. One main drawback is that they require huge amounts of computational resources. This drawback is much obvious for an autonomous mobile robot that must carry its own computational resources. Secondly, this approach must be based on highly accurate model, thus it requires a number of high-precision sensors which are also often expensive. These sensors, however, are subject to noisy data. Finally, this sense-plan-act paradigm is by nature sequential, thus it would fail if the world happens to change in between of phases. Furthermore, there is always delay between sensing and act, due to longer time required in planning.

As an alternative to the traditional approach, a new paradigm called subsumption architecture, also known as behaviour-based control, is devised [17]. In this architecture, sensors are dealt with only implicitly in that they initiate behaviours. Each behaviour is simply layers of control systems that all run in parallel. Higher level behaviours have the power to temporarily suppress lower level behaviours. Therefore, a set of priority scheme is used to resolve the dominant behaviour for a given scenario. A more rigorous explanation of behaviour-based approach for controlling robot is presented in [18]. In this work, inspired by the behaviour-based paradigm approach to robotics control, the searching algorithm employs Laplacian Behaviour-Based Control (LBBC) for robust space exploration of the configuration space. The LBBC comprises four core behaviours i.e. *keep-forward*, *follow-wall*, *avoid-obstacle*, and *find-slope*. All these core behaviours make use of the potential values represented by temperature distribution in the configuration space which are computed numerically to provide guidance during search exploration.

A. Keep-Forward Behaviour

The *keep-forward* behaviour is a core behaviour that keeps the searching moving forward in the same direction as long as the temperature at current location is higher than the next location. When the searching encounters ascending slope, flat region, obstacles or walls, the *keep-forward* behaviour stops, and other behaviours would take over. The main aim of this behaviour is to guide the searching by following the descending slope until the goal location is found.

B. Follow-Wall Behaviour

The *follow-wall* behaviour provides the search with the capability to follow the wall for a specified number of steps. With this behaviour, it will command the searching to keep turning gradually until its direction is parallel with the wall. It provides the searching with the capability of traversing the narrow path and sharp corner. In this implementation, the *follow-wall* behaviour is executed for every a specified number of steps. After that the searching switches to *find-slope* behaviour.

C. Avoid-Obstacle Behaviour

When the searching hits an obstacle or wall, it will trigger the searching to backup and turn 90 degrees to the left or right alternately. By turning alternately to the left and right, it provides the searching with the capability to escape from a difficult position such as sharp corner.

D. Find-Slope Behaviour

When the *find-slope* behaviour takes over, it will command the searching to move randomly hoping to encounter a descending slope that consequently triggers *keep-forward* behaviour. With this behaviour, the searching is capable of moving away from a flat region to continue its descending move towards goal location.

4. Harmonic Functions

A harmonic function on a domain is a function which satisfies Laplace's equation, see Eq. (1), where x_i is the *i*-th Cartesian coordinate and *n* is the dimension. In the case of robot path construction, the boundary of Ω (denoted by $\partial\Omega$) consists of the outer boundary of the workspace and the boundaries of all the obstacles as well as the start point and the goal point, in a configuration space representation. The spontaneous creation of a false local minimum inside the region Ω is avoided if Laplace's equation is imposed as a constraint on the functions used, as the harmonic functions satisfy the min-max principle. Laplace's equation can be solved numerically. Standard methods are Jacobi and Gauss-Seidel. Faster computation can be achieved via weighted iterative method using Successive-Over-Relaxation (SOR). In this paper, a new iteration method using Full-Sweep Successive-Over-Relaxation via Nine-Point Laplacian (FSSOR9L) is considered.

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

5. Configuration Space

In the framework used in this study, the robot is represented by a point in the configuration space, or C-space. The path planning problem is then posed as an obstacle avoidance problem for the point robot from the start point to the goal point in the C- space. The C-space can have either square or rectangular outer boundaries, having projections or convolutions inside to act as barriers. Apart from projections of the boundaries, some obstacles inside the boundary are also considered. The C-space is designed in grid or discrete form and the coordinates and function values associated with each node are computed iteratively by applying numerical technique to satisfy equation in Eq. (1). The highest temperature is assigned to the start point whereas the goal point is assigned the lowest. In some cases with Dirichlet conditions, the start point is not assigned any temperature. In this study, Dirichlet boundary conditions are employed, thus the results are processed by assigning different temperature values to the boundaries and obstacles, and lowest temperature for the goal point. No temperature values are assigned to the start points. In this work, solution to the Laplace's equation were subjected to Dirichlet boundary conditions, $\Phi \mid \partial\Omega = c$, where c is constant.

6. Full-Sweep Successive Over-Relaxation via Nine-Point Laplacian (FSSOR9L) Iterative Method

In the literature, Jacobi method [9] and Gauss-Seidel method [7] had been used for solving any linear system. Daily and Bevly [11] use analytical solution for arbitrarily shaped obstacles. Block iterative method was discussed in [2], [3], [4] and [10]. In this study, the computation employs a weighted iterative method for solving the Laplace's equation. Let us consider the two-dimensional Laplace equation in Eq. (1) defined as

$$\frac{\partial^2 U}{\partial^2 x} + \frac{\partial^2 U}{\partial^2 y} = 0 \tag{2}$$

The discretization of Eq. (2) based on the 9-point Laplacian can be shown as below

$$\begin{aligned}
&4(U_W + U_E + U_N + U_S) + D_{BL} + D_{BR} + D_{TL} + D_{TR} - 20U_C = 0,\\
&W = i-1,\ j;\ E = i+1;\ N = i,\ j+1;\ S = i,\ j-1;\\
&BL = i-1,\ j-1;\ BR = i+1,\ j-1;\\
&TL = i-1,\ j+1;\ TR = i+1,\ j+1;\\
&C = i,\ j
\end{aligned} \tag{3}$$

Based on Eq. (3), the standard Gauss-Seidel iterative method for solving linear system can be shown as

$$U_C = \frac{1}{20}\left[4(U_W + U_E + U_N + U_S) + D_{BL} + D_{BR} + D_{TL} + D_{TR}\right] \quad (4)$$

Then the formulation of the SOR method can be shown as follows (Young [21]):

$$U_C^{k+1} = \frac{\omega}{20}\left[4(U_W^k + U_E^k + U_N^k + U_S^k) + D_{BL}^k + D_{BR}^k + D_{TL}^k + D_{TL}^k\right] + (1-\omega)U_C^k \quad (5)$$

where the optimal value of ω is defined in the range $1 \le \omega < 2$. In order to find the optimal value, several runs of computer program implementation of Eq. (5) were carried out with varying value of ω. The value of ω is considered optimal when the program converges with the less number of iterations. By taking $\omega = 1$, the SOR iterative method will represent Gauss-Seidel method. As can be seen in Eq. (5), the SOR iterative method can be categorized as a family of point iterative methods.

7. Experiments and Results

The experiment considered various size of static environment, i.e. 64x64, 128x128, 256x256, and 512x512, that consists of a goal point, several starting points and varying setup of walls. Initially, the outer and inner walls were fixed with high temperature values. Goal point was set to very low temperature. All other free spaces were set to zero temperature value.

A. Computation of Temperature Values

The iteration process was run on Intel Core 2 Duo CPU running at 1.83GHz speed with 1GB of RAM to compute temperature values numerically at all points in the environment. The iteration process was terminated when there was no more changes in temperature values, where it converged to a specified very small value, i.e. 1.0^{-10}. The highest precision of solution for Eq. (1) was required to reduce the occurrence of flat area, hence would speed up the searching algorithm during path planning construction of the mobile robot from starting point to goal point. Table I shows the number of iterations, maximum error and CPU time (in seconds) required to compute all temperature values in the environment. Clearly, FSSOR9L iterative method proved to be very fast compared to the standard Gauss-Seidel, and slightly faster than the standard SOR implementation. As the size of environment gets bigger, the amount of computation required increased exponentially for all iterative methods, especially for Gauss-Seidel iteration. For 512x512 environment, FSSOR9L is 18 times faster than Gauss-Seidel and it reduces the number of iteration by 16% against the standard SOR implementation.

B. Path Planning Construction

Once the temperature values were obtained, the searching algorithm would make use of them to guide its exploration. In the previous work, the path can be generated successfully even without LBBC, if the environment space was simple and sparse in which the gradient from start points to goal point are smooth, as shown in Figure 2. However, the searching algorithm failed to reach the goal point when the horizontal wall was extended. As shown in Figure 3, only one path was successfully generated, whereas the other two start points got stuck in the flat region. By employing LBBC, the searching would be able to escape from flat region and continue its exploration by utilizing *follow-wall* behaviour to reach the goal point, see Figure 4. As shown in Figure 5, the temperature values of the walls and the generated paths are raised up for visualization purpose. The lowest temperature indicates the goal point. All other areas are almost flat due to very small difference in temperature values, except for the area close to the goal point.

8. Conclusions

The experiment in this study shows the effectiveness of the Laplacian Behaviour-Based Control (LBBC) in generating path for mobile robot in varying setup of complex environment. Unlike previous methods [19] and [20], the LBBC provides the searching algorithm with the capability to escape from flat region and difficult position, thus the searching algorithm could continue its move towards goal location. Moreover, the potential values of each point in the configuration, i.e. the solution of Laplace's equation, as shown in Eq. (1), can be computed rapidly by using weighted iterative method. As the size of configuration space gets bigger, however, as shown in Table 1 and Figure 1, the CPU time required increased tremendously, thus impractical for fast and real time robotics applications. Future work would include the implementation of block iterative method to speed up the computation of solving the Laplace's equation. The application of this LBBC technique via Laplace's equation for other industrial robotics would also be considered.

References

[1] Connolly, C. I., & Gruppen, R. 1993. On the applications of harmonic functions to robotics. Journal of Robotic Systems, 10(7): 931–946.

[2] Evans, D. J. 1985. Group Explicit Iterative methods for solving large linear systems. Int. J. Computer Maths., 17: 81-108.

[3] Evans, D.J & Yousif, W. S. 1986. Explicit Group Iterative Methods for solving elliptic partial differential equations in 3-space dimensions. Int. J. Computer Maths., 18:323-340.

[4] Ibrahim, A. 1993. The Study of the Iterative Solution of Boundary Value Problem by the Finite Difference Methods. PhD Thesis. Universiti Kebangsaan Malaysia.

[5] Khatib, O. 1985. Real time obstacle avoidance for manipulators and mobile robots. IEEE Transactions on Robotics and Automation 1:500–505.

[6] 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: 1-6.

[7] Connolly, C. I., Burns, J. B., & Weiss, R. 1990. Path planning using Laplace's equation. Proceedings of the IEEE International Conference on Robotics and Automation: 2102–2106.

[8] Akishita, S., Kawamura, S., & Hayashi, K. 1990. Laplace potential for moving obstacle avoidance and approach of a mobile robot. Japan-USA Symposium on flexible automation, A Pacific rim conference: 139–142.

[9] Sasaki, S. 1998. A Practical Computational Technique for Mobile Robot Navigation. Proceedings of the IEEE International Conference on Control Applications: 1323-1327.

[10] Sulaiman, J., Hasan, M.K. & Othman, M. 2007. Red-Black EDGSOR Iterative Method Using Triangle Element Approximation for 2D Poisson Equations. In. O. Gervasi & M. Gavrilova (Eds). Computational Science and Its Application 2007. Lecture Notes in Computer Science (LNCS 4707): 298-308. Berlin: Springer-Verlag.

[11] Daily, R., & Bevly, D.M. 2008. Harmonic Potential Field Path Planning for High Speed Vehicles. In the proceeding of American Control Conference, Seattle, June 11-13, 4609-4614.

[12] Willms, A.R. and Simon X.Y. 2008. IEEE Trans. on Systems, Man, and Cybernetics – Part B: Cybernetics, Vol. 38. No. 3, June 2008. Real-Time Robot Path Planning via a Distance-Propagating Dynamic System with Obstacle Clearance.

[13] Jan. G.E., Chang, K.Y., and Parberry I. 2008. Optimal Path Planning for Mobile Robot Navigation. IEEE/ASME Trans. on Mechatronics, Vol. 13, No. 4, Aug 2008, pages 451-460.

[14] Bhattacharya, P. and Gavrilova, M.L. 2008. Roadmap-Based Path Planning - Using the Voronoi Diagram for a Clearance-Based Shortest Path. IEEE Robotics & Automation Magazine, Volume 15, Issue 2, June 2008. Page(s):58 – 66.

[15] Chen, W. And Qin, H. 2011. Path planning of mobile robot based on Hybrid Cascaded Genetic Algorithm. In proc. of the 9th World Congress on Intelligent Control and Automation (WCICA), 21-25 June 2011. Page(s): 501 – 504. ISBN: 978-1-61284-698-9.

[16] R. A. Brooks. 1986. A robust layered control system for a mobile robot, IEEE Journal of Robotics and Automation, 2:(14-23).

[17] R. C. Arkin. 1998. Behaviour-based robotics, Bradford Books.

[18] Saudi, A. and Hallam, B. 2004. A Tale of Two Behaviour-based Robots: Mindstorms vs Technic. In proc. of IEEE Region 10 Conference (TENCON 2004), 21-24 Nov. 2004. Page(s): 479 - 482 Vol. 4. ISBN: 0-7803-8560-8.

[19] Saudi, A. and Sulaiman, J. 2009. Efficient Weighted Block Iterative Method for Robot Path Planning Using Harmonic Functions. In proc. of the Second International Conference on Control, Instrumentation and Mechatronic Engineering (CIM2009), Malacca, Malaysia, June 2-3, 2009.

[20] Saudi, A. and Sulaiman, J. 2010. Robot Path Planning Based On Four Point-EGSOR Iterative Method. In proc. of the 4th IEEE Int. Conf. on Robotics, Automation and Mechatronics (RAM 2010), 28 – 30 June, 2010, Singapore. Page(s): 476 – 481. ISBN: 978-1-4244-6503.

[21] Young, D.M. 1971. Iterative solution of large linear systems. London: Academic Press.

Table I: Performance comparison of Several Iterative Methods in Varying Size of Environment.

		Size of environment			
		64x64	**128x128**	**256x256**	**512x512**
Number of iterations	***Gauss-Seidel***	3898	14218	51363	NA
	SOR	233	836	3265	12071
	FSSOR9L	232	701	2750	10184
Maximum error	***Gauss-Seidel***	0.9988^{-10}	0.9995^{-10}	0.9998^{-10}	NA
	SOR	0.9770^{-10}	0.9800^{-10}	0.9981^{-10}	0.9993^{-10}
	FSSOR9L	0.8543^{-10}	0.9778^{-10}	0.9964^{-10}	0.9989^{-10}
CPU time (s)	***Gauss-Seidel***	7	103	1523	NA
	SOR	0.5	6	97	1429
	FSSOR9L	0.5	5	87	1293

Note: NA – The performance for Gauss-Seidel against size 512x512 is not included since the computional resourses required is too huge and too impractical to implement.

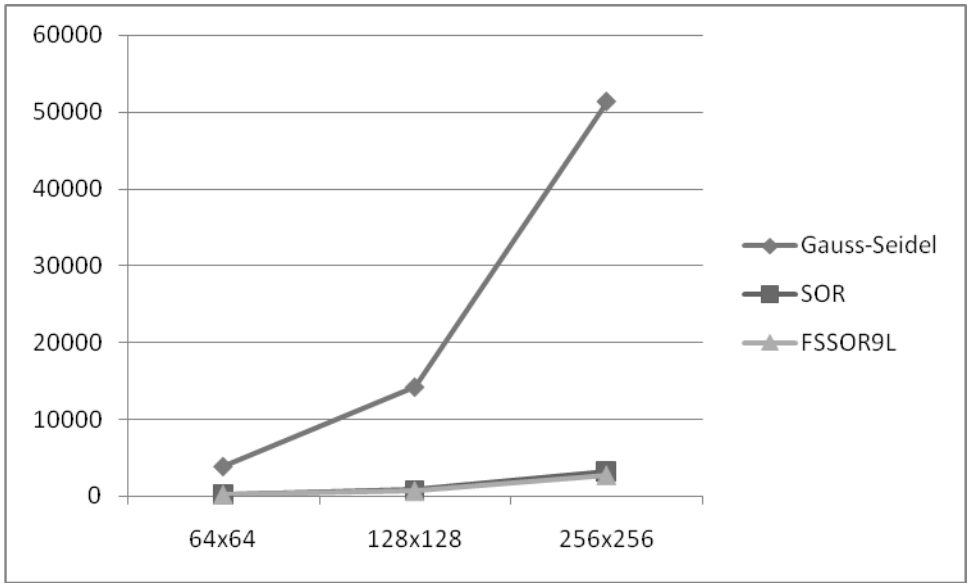

Figure 1: The graph of number of iteration for various iterative methods against varying sizes of environment.

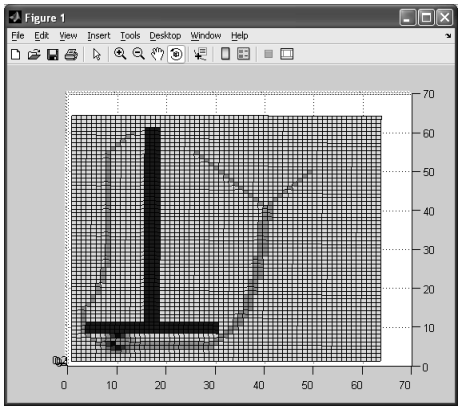

Figure 2: Path is successfully generated in a simple and sparse environment.

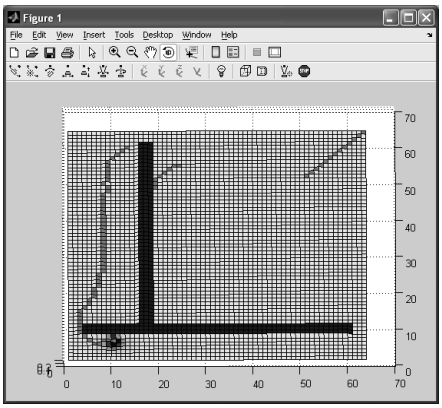

Figure 3: The path generation process failed to reach the goal point when the length of horizontal wall is extended twice to the right.

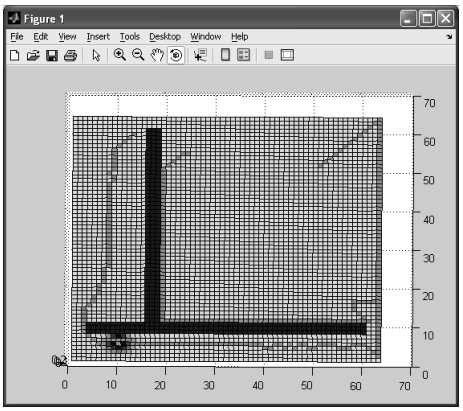

Figure 4: With LBBC, the algorithm simply utilized the follow-wall behaviour to escape from flat region and keep moving to find the goal point.

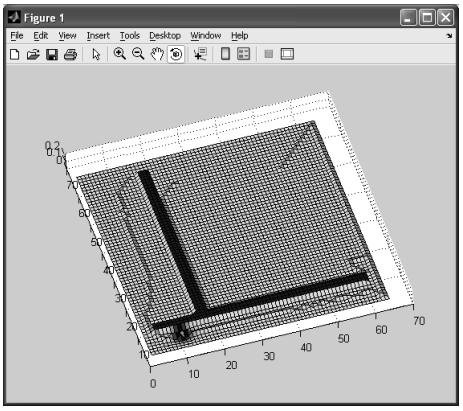

Figure 5: The 3D view of the environment and generated path from three start points to a goal point.