

**ROBOT PATH PLANNING
USING FAMILY OF SOR ITERATIVE METHODS
WITH LAPLACIAN BEHAVIOUR-BASED CONTROL**

AZALI BIN SAUDI



UMS
UNIVERSITI MALAYSIA SABAH

**THESIS SUBMITTED IN FULFILLMENT FOR THE
DEGREE OF DOCTOR OF PHILOSOPHY**

**FACULTY OF SCIENCE AND NATURAL RESOURCES
UNIVERSITI MALAYSIA SABAH
2015**

BORANG PENGESAHAN STATUS TESIS

JUDUL: ROBOT PATH PLANNING USING FAMILY OF SOR ITERATIVE METHODS
WITH LAPLACIAN BEHAVIOUR-BASED CONTROL

IJAZAH: DOKTOR FALSAFAH dalam bidang PENGKOMPUTERAN SAINTIFIK

Saya AZALI BIN SAUDI, Sesi Pengajian 2009-2015, mengaku membenarkan tesis
Doktor Falsafah ini disimpan di Perpustakaan Universiti Malaysia Sabah dengan
syarat-syarat kegunaan seperti berikut:

1. Tesis ini adalah hak milik Universiti Malaysia Sabah.
2. Perpustakaan Universiti Malaysia Sabah dibenarkan membuat salinan untuk tujuan pengajian sahaja.
3. Perpustakaan dibenarkan membuat salinan tesis ini sebagai bahan pertukaran antara institusi pengajian tinggi.
4. Sila tandakan (/)

☐ SULIT (Mengandungi maklumat berdarjah keselamatan atau kepentingan Malaysia seperti yang termaktub di dalam AKTA RAHSIA RASMI 1972)

☐ TERHAD (Mengandungi maklumat TERHAD yang telah ditentukan oleh organisasi/badan di mana penyelidikan dijalankan)

☒ TIDAK TERHAD

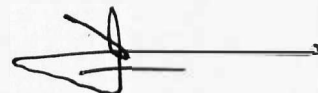


(Tandatangan Penulis)

Disahkan oleh,
NORAZLYNNE MOHD. JOHAN @ JACKLYNE
PUSTAKAWAN
UNIVERSITI MALAYSIA SABAH

(Tandatangan Pustakawan)

Alamat Tetap: Hse 15, Lorong 4/1B,
Bandar Sierra, Menggatal,
88450 Kota Kinabalu,
Sabah.



Tarikh: 28 Ogos 2015

(PROF. MADYA DR. JUMAT SULAIMAN)
Penyelia

DECLARATION

I hereby declare that the material in this thesis is my own except for quotations, excerpts, equations, summaries and references, which have been duly acknowledged.

27th August 2015



Azali Bin Saudi
PS20099014



UMS
UNIVERSITI MALAYSIA SABAH

CERTIFICATION

NAME : **AZALI BIN SAUDI**

MATRIC NO : **PS20099014**

TITLE : **ROBOT PATH PLANNING USING FAMILY OF SOR ITERATIVE
METHODS WITH LAPLACIAN BEHAVIOUR-BASED CONTROL**

DEGREE : **DOCTOR OF PHILOSOPHY
(SCIENTIFIC COMPUTING)**

VIVA DATE : **27th July 2015**

DECLARED BY

1. MAIN SUPERVISOR

Associate Professor Dr. Jumat Sulaiman



Signature

A handwritten signature in black ink, consisting of a large, stylized 'J' followed by a horizontal line and a small loop.

2. CO-SUPERVISOR

Dr. Mohd. Hanafi Ahmad Hijazi

A handwritten signature in black ink, featuring a large, stylized 'H' followed by a vertical line and a small loop.

UMS
UNIVERSITI MALAYSIA SABAH

ACKNOWLEDGEMENT

First and foremost, I thank my supervisor Associate Professor Dr. Jumat Sulaiman for the invaluable guidance and constant encouragement he has given me throughout the study. Without his enthusiasm, comments and patience when listening to my sometimes confused ideas, this work would not have been completed.

I also express my gratitude to my co-supervisor Dr. Mohd. Hanafi Ahmad Hijazi for his comments and guidance particularly during the writing of this thesis.

I would also like to express my love and gratitude to my parents, for their understanding and patience throughout my study. I wish to thank my wife Suzanty and our children Nur Aqilah, Muhammad Zuhair and Muhammad Zakwan for their love during this study, and for being an unlimited source of joy and inspiration.

Finally, I acknowledge Universiti Malaysia Sabah and Skim Latihan Akademik IPTA, Ministry of Higher Education for the financial supports.



UMS
UNIVERSITI MALAYSIA SABAH

ABSTRACT

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 potential values. It is based on the theory of heat transfer, when there exist a temperature gradient within a surface, heat energy will flow from the region of high temperature at heat source to the region of low temperature at heat sink. In this model, high Laplacian potentials are assigned to outer boundary, inner walls and obstacles. Whilst, the goal point is assigned the lowest and no Laplacian potentials are assigned to all other free spaces. The Laplacian potentials for nodes on free spaces are then computed iteratively using numerical techniques. In the literature, computing these Laplacian potentials using numerical techniques produced encouraging results. The numerical implementations of these previous works, however, were only based on family of point iterative methods i.e. Jacobi, Gauss-Seidel and Successive Overrelaxation (SOR). These standard methods are too slow when handling large environment. Therefore, this study introduces the concepts of half-sweep and quarter-sweep iterations, and initiates the first application of using family of Point SOR and family of Four Point-Block SOR iterative methods for computing the Laplacian potentials to solve the path planning problem. The implementations employ two finite difference discretization schemes that are based on 5-Point and 9-Point Laplacian. Within the family of Point SOR iterative methods, the simulation results shows that the application of half-sweep and quarter-sweep concepts reduced the computational complexities of the algorithms by approximately 50% and 75%, respectively. Significantly, simulations with family of Four Point-Block SOR iterative methods provide even faster computation. In terms of iterations count, the iterative methods based on the 9-Point Laplacian give the less number of iterations than the 5-Point Laplacian. Whilst, in terms of execution time, the speed difference between iterative methods based on 5-Point and 9-Point Laplacian is very minimal. Once the Laplacian potentials are obtained, the standard Gradient Descent Search (GDS) technique is performed for path tracing to the goal point. The existing GDS, however, suffers from the occurrence of flat region in a more difficult environment which causing the path generation to fail. Thus, this study proposes a new control known as Laplacian Behaviour-Based Control (LBBC) to overcome such problem. Due to its robustness, the LBBC successfully generated smooth path even in a more complex configuration space. Therefore in conclusion, the significant contribution of this study is in introducing for the first time the fast half-sweep and quarter-sweep iterative methods using families of Point SOR and Four Point-Block SOR methods via 5-Point and 9-Point Laplacian. These faster iterative methods overcome the slow performances of the existing standard methods, particularly when handling large environment. In addition, the newly proposed LBBC overcomes the drawback of the existing GDS that face difficulty when handling complex environment. Finally, the path planning problem is solved by combining the fast iterative method with the robust path searching LBBC technique, so that the path planning algorithm is capable of handling large and complex environment.

PERANCANGAN LALUAN ROBOT MENGGUNAKAN FAMILI KAEDAH LELARAN SOR DENGAN KAWALAN BERASASKAN-KELAKUAN LAPLACIAN

Robot automatik yang sebenar perlulah berkeupayaan untuk mencari laluan dari titik permulaan hingga ke titik destinasi. Kajian ini mencadangkan teknik perancangan laluan robot yang menggunakan persamaan Laplace untuk menjana nilai-nilai potensi. Ianya berdasarkan teori pemindahan haba, apabila terdapat kecerunan suhu pada permukaan, haba akan mengalir dari kawasan sumber suhu yang bersuhu tinggi ke kawasan bersuhu rendah yang bertindak sebagai penarik suhu. Dengan model ini, nilai potensi Laplacian yang tinggi diberikan kepada dinding luar, dinding dalaman dan objek halangan. Manakala titik destinasi diberikan nilai paling rendah, dan tiada nilai potensi Laplacian diberikan kepada titik-titik bebas yang lain. Nilai potensi Laplacian untuk titik-titik bebas kemudiannya akan dihitung secara lelaran menggunakan teknik berangka. Dalam kajian lepas, pengiraan nilai-nilai potensi Laplacian dengan menggunakan teknik berangka menghasilkan keputusan yang menggalakkan. Bagaimanapun, implementasi teknik berangka dalam kajian-kajian yang lepas ini hanya berasaskan kaedah lelaran titik iaitu Jacobi, Gauss-Seidel dan Successive Overrelaxation (SOR). Kaedah-kaedah lelaran lazim ini terlalu perlahan apabila digunakan untuk persekitaran yang luas. Oleh yang demikian, kajian ini memperkenalkan konsep lelaran sapuan-separuh dan sapuan-suku dengan buat pertama kali mengaplikasikan penggunaannya melalui kaedah lelaran family of Point SOR dan family of Four Point-Block SOR untuk menghitung potensi-potensi Laplacian bagi menyelesaikan masalah perancangan laluan. Dalam implementasinya, dua skema pendiskretan pembezaan terhitung digunakan yang berasaskan 5-Point dan 9-Point Laplacian. Dengan kaedah lelaran family of Point SOR, keputusan simulasi menunjukkan aplikasi sapuan-separuh dan sapuan-suku telah mengurangkan kekompleksan pengiraan algoritma masing-masing sekitar 50% dan 75%. Manakala, kaedah lelaran family of Four Point-Block SOR pula telah menyediakan pengiraan yang lebih pantas. Dari segi bilangan lelaran, kaedah berasaskan 9-Point Laplacian memberikan bilangan lelaran lebih rendah berbanding kaedah 5-Point Laplacian. Manakala, dari segi masa pelaksanaan, perbezaan kepantasan antara kaedah 5-Point dan 9-Point Laplacian adalah sangat minimum. Setelah nilai-nilai potensi Laplacian diperolehi, teknik Gradient Descent Search (GDS) digunakan untuk menjejak laluan ke titik destinasi. Namun, teknik GDS mengalami masalah apabila terdapat kawasan rata dalam persekitaran yang lebih sukar dan menyebabkan penjanaan laluan gagal. Oleh itu, kajian ini mencadangkan teknik kawalan baru yang dikenali sebagai Laplacian Behaviour-Based Control (LBBC) bagi mengatasi masalah tersebut. Teknik LBBC telah berjaya menjana laluan yang lancar walaupun pada ruang konfigurasi yang kompleks. Sebagai kesimpulan, sumbangan terpenting kajian ini ialah memperkenalkan buat pertama kali lelaran sapuan-separuh dan sapuan-suku dalam kaedah lelaran family of Point SOR dan family of Four Point-Block SOR dengan berasaskan 5-Point dan 9-Point Laplacian. Kaedah-kaedah lelaran yang laju ini mengatasi kaedah sedia ada yang terlalu perlahan, terutamanya untuk persekitaran yang luas. Selain itu, teknik baru LBBC dapat mengatasi kelemahan teknik sedia ada GDS yang mengalami kesukaran apabila mengendalikan persekitaran yang kompleks. Seterusnya, masalah perancangan laluan diselesaikan dengan menggabungkan kaedah lelaran yang laju dengan teknik carian laluan LBBC yang cekap, lantas algoritma mampu mengendalikan persekitaran yang luas dan kompleks.

LIST OF CONTENTS

	Page
TITLE	i
DECLARATION	ii
CERTIFICATION	iii
ACKNOWLEDGMENT	iv
ABSTRACT	v
ABSTRAK	vi
TABLE OF CONTENTS	vii
LIST OF TABLES	viii
LIST OF FIGURES	ix
LIST OF ABBREVIATIONS	x
LIST OF SYMBOLS	xii
CHAPTER 1: INTRODUCTION	1
1.1 Background	1
1.2 Path Planning Problem	4
1.2.1 Local Path Planning	5
1.2.2 Global Path Planning	5
1.3 Path Planning using Laplace's Equation	6
1.3.1 Iterative Methods	7
1.4 Problem Statement	10
1.5 Research Questions	11
1.6 Significance of Research	11
1.7 Objectives of Study	13
1.8 Scope and Restrictions of Study	13
1.9 Outline of the Thesis	14
CHAPTER 2: LITERATURE REVIEW	18
2.1 Introduction	18
2.2 Path Planning for Mobile Robot	18
2.2.1 Local Path Planning	24
2.2.2 Global Path Planning	25

2.2.3	Gradient Descent Search	27
2.3	Applications of Laplace's Equation in Robotics	28
2.4	Numerical Methods for Laplace's Equation	29
2.4.1	Iterative Methods for Linear System	30
2.4.2	Classifications of Iterative Methods	31
2.4.3	Complexity Reduction Approach	35
2.5	Robot Control Architectures	35
2.5.1	Deliberative Strategy	36
2.5.2	Reactive Approach	37
2.5.3	Hybrid Architecture	38
2.5.4	Behaviour-Based Approach	38
2.6	Path Planning Strategy	40
2.6.1	Physical Analogy	41
2.6.2	Harmonic Function	42
2.6.3	Configuration Space	43
2.6.4	Path Generation	44
2.6.5	Robot Simulator	44
2.7	Half- and Quarter-Sweep Iteration Concepts	49
2.8	Research Motivations	50
CHAPTER 3: THE ITERATIVE METHODS AND PATH SEARCHING TECHNIQUES FOR SOLVING PATH PLANNING PROBLEM		52
3.1	Introduction	52
3.2	Iterative Methods for Solving Laplace's equation	52
3.3	The Five-Point Stencil for the Laplacian (5L)	53
3.3.1	Five-Point Finite Difference Approximations	59
3.4	The Nine-Point Laplacian (9L)	65
3.4.1	Nine-Point Finite Difference Approximations	69
3.5	Formulation of Family of Point SOR Methods via 5L	75
3.6	Formulation of Family of Four Point-Block SOR Methods via 5L	81
3.6.1	Four Point-EGSOR Method via 5L	81

3.6.2	Four Point-EDGSOR Method via 5L	85
3.6.3	Four Point-MEGSOR Method via 5L	88
3.7	Formulation of Family of Point SOR Methods via 9L	91
3.8	Formulation of Family of Four Point-Block SOR Methods via 9L	97
3.8.1	Four Point-EGSOR Method via 9L	97
3.8.2	Four Point-EDGSOR Method via 9L	100
3.8.3	Four Point-MEGSOR Method via 9L	104
3.9	Searching Techniques for Path Generation	107
3.9.1	Gradient Descent Search (GDS)	107
3.9.2	Behaviour-Based Paradigm	109
3.9.3	Laplacian Behaviour-Based Control (LBBC)	110
3.10	Path Planning Algorithm using Iterative Methods with GDS	120
3.11	Path Planning Algorithm using Iterative Methods with LBBC	125
CHAPTER 4: PATH PLANNING USING ITERATIVE METHODS WITH GRADIENT DESCENT SEARCH (GDS)		130
4.1	Introduction	130
4.2	Simulation using Family of Point SOR Methods via 5-Point Laplacian with GDS	131
4.2.1	Simulation Results and Discussions	132
4.3	Simulation using Family of Four Point-Block SOR Methods via 5-Point Laplacian with GDS	156
4.3.1	Simulation Results and Discussions	156
4.4	Simulation using Family of Point SOR Methods via 9-Point Laplacian with GDS	176
4.4.1	Simulation Results and Discussions	176
4.5	Simulation using Family of Four Point-Block SOR Methods via 9-Point Laplacian with GDS	196
4.5.1	Simulation Results and Discussions	196
4.6	Analysis of Computational Complexity	216
4.7	Concluding Remarks	217

CHAPTER 5: PATH PLANNING USING ITERATIVE METHODS WITH	225
LAPLACIAN BEHAVIOUR-BASED CONTROL (LBBC)	
5.1 Introduction	225
5.2 Simulation using Family of Point SOR Methods via 5-Point Laplacian with LBBC	225
5.2.1 Simulation Results and Discussion	226
5.3 Simulation using Family of Four Point-Block SOR Methods via 5-Point Laplacian with LBBC	250
5.3.1 Simulation Results and Discussion	250
5.4 Simulation using Family of Point SOR Methods via 9-Point Laplacian with LBBC	274
5.4.1 Simulation Results and Discussion	274
5.5 Simulation using Family of Four Point-Block SOR Methods via 9-Point Laplacian with LBBC	298
5.5.1 Simulation Results and Discussion	298
5.6 Analysis of Computational Complexity	322
5.7 Concluding Remarks	323
CHAPTER 6: CONCLUSION	329
6.1 Summary of the Study	329
6.2 Conclusions	330
6.3 Recommendation of Future Research	332
REFERENCES	334
LIST OF PUBLICATIONS	350

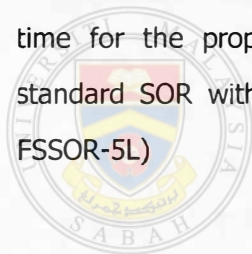
LIST OF TABLES

Table 4.1	Performance in terms of number of iterations for Family of Point SOR methods via 5L with GDS	141
Table 4.2	Performance in terms of CPU time (in seconds) for Family of Point SOR methods via 5L with GDS	142
Table 4.3	Maximum absolute error of Family of Point SOR methods via 5L with GDS	143
Table 4.4	Reduction percentages in terms of number of iterations and CPU time for GDS-HSSOR-5L and GDS-QSSOR-5L compared with GDS-FSSOR-5L	144
Table 4.5	Performance in terms of number of iterations for Family of Four Point-Block SOR methods via 5L with GDS	165
Table 4.6	Performance in terms of CPU time (in seconds) for Family of Four Point-Block SOR methods via 5L with GDS	166
Table 4.7	Maximum absolute error of Family of Four Point-Block SOR methods via 5L with GDS	167
Table 4.8	Reduction percentages in terms of number of iterations and CPU time for GDS-4-EDGSOR-5L and GDS-4-MEGSOR-5L compared with GDS-4-EGSOR-5L	168
Table 4.9	Performance in terms of number of iterations for Family of Point SOR methods via 9L with GDS	185
Table 4.10	Performance in terms of CPU time (in seconds) for Family of Point SOR methods via 9L with GDS	186
Table 4.11	Maximum absolute error of Family of Point SOR methods via 9L with GDS	187
Table 4.12	Reduction percentages in terms of number of iterations and CPU time for GDS-HSSOR-9L and GDS-QSSOR-9L compared with GDS-FSSOR-9L	188
Table 4.13	Performance in terms of number of iterations for Family of Four Point-Block SOR methods via 9L with GDS	205

Table 4.14	Performance in terms of CPU time (in seconds) for Family of Four Point-Block SOR methods via 9L with GDS	206
Table 4.15	Maximum absolute error of Family of Four Point-Block SOR methods via 9L with GDS	207
Table 4.16	Reduction percentages in terms of number of iterations and CPU time for GDS-4-EDGSOR-9L and GDS-4-MEGSOR-9L compared with GDS-4-EGSOR-9L	208
Table 4.17	Number of arithmetic operations per iteration for FSSOR-5L, HSSOR-5L and QSSOR-5L methods	220
Table 4.18	Number of arithmetic operations per iteration for 4-EGSOR-5L, 4- EDGSOR-5L and 4-MEGSOR-5L methods	220
Table 4.19	Number of arithmetic operations per iteration for FSSOR-9L, HSSOR-9L and QSSOR-9L methods	220
Table 4.20	Number of arithmetic operations per iteration for 4-EGSOR-9L, 4- EDGSOR-9L and 4-MEGSOR-9L methods	221
Table 4.21	Number of arithmetic operations to calculate the remaining points using direct methods	221
Table 4.22	Total number of arithmetic operations for path planning algorithm using family of Point SOR methods via 5L with GDS	222
Table 4.23	Total number of arithmetic operations for path planning algorithm using family of Four Point-Block SOR methods via 5L with GDS	222
Table 4.24	Total number of arithmetic operations for path planning algorithm using family of Point SOR methods via 9L with GDS	223
Table 4.25	Total number of arithmetic operations for path planning algorithm using family of Four Point-Block SOR methods via 9L with GDS	223
Table 4.26	Reduction percentages in terms of number of iterations and CPU time for the proposed algorithms compared to the existing standard SOR with GDS technique (also now known as GDS-FSSOR-5L)	224

Table 5.1	Performance in terms of number of iterations for Family of Point SOR methods via 5L with LBBC	235
Table 5.2	Performance in terms of CPU time (in seconds) for Family of Point SOR methods via 5L with LBBC	236
Table 5.3	Maximum absolute error of Family of Point SOR methods via 5L with LBBC	237
Table 5.4	Reduction percentages in terms of number of iterations and CPU time for LBBC-HSSOR-5L and LBBC-QSSOR-5L compared with LBBC-FSSOR-5L	238
Table 5.5	Performance in terms of number of iterations for Family of Four Point-Block SOR methods via 5L with LBBC	259
Table 5.6	Performance in terms of CPU time (in seconds) for Family of Four Point-Block SOR methods via 5L with LBBC	260
Table 5.7	Maximum absolute error of Family of Four Point-Block SOR methods via 5L with LBBC	261
Table 5.8	Reduction percentages in terms of number of iterations and CPU time for LBBC-4-EDGSOR-5L and LBBC-4-MEGSOR-5L compared with LBBC-4-EGSOR-5L	262
Table 5.9	Performance in terms of number of iterations for Family of Point SOR methods via 9L with LBBC	283
Table 5.10	Performance in terms of CPU time (in seconds) for Family of Point SOR methods via 9L with LBBC	284
Table 5.11	Maximum absolute error of Family of Point SOR methods via 9L with LBBC	285
Table 5.12	Reduction percentages in terms of number of iterations and CPU time for LBBC-HSSOR-9L and LBBC-QSSOR-9L compared with LBBC-FSSOR-9L	286
Table 5.13	Performance in terms of number of iterations for Family of Four Point-Block SOR methods via 9L with LBBC	307
Table 5.14	Performance in terms of CPU time (in seconds) for Family of Four Point-Block SOR methods via 9L with LBBC	308

Table 5.15	Maximum absolute error of Family of Four Point-Block SOR methods via 9L with LBBC	309
Table 5.16	Reduction percentages in terms of number of iterations and CPU time for LBBC-4-EDGSOR-9L and LBBC-4-MEGSOR-9L compared with LBBC-4-EGSOR-9L	310
Table 5.17	Number of arithmetic operations for LBBC-FSSOR-5L, LBBC-HSSOR-5L and LBBC-QSSOR-5L methods	326
Table 5.18	Number of arithmetic operations for LBBC-4-EGSOR-5L, LBBC-4-EDGSOR-5L and LBBC-4-MEGSOR-5L methods	326
Table 5.19	Number of arithmetic operations for LBBC-FSSOR-9L, LBBC-HSSOR-9L and LBBC-QSSOR-9L methods	327
Table 5.20	Number of arithmetic operations for LBBC-4-EGSOR-9L, LBBC-4-EDGSOR-9L and LBBC-4-MEGSOR-9L methods	327
Table 5.21	Reduction percentages in terms of number of iterations and CPU time for the proposed algorithms compared to the existing standard SOR with GDS technique (also now known as GDS-FSSOR-5L)	328



UMS
UNIVERSITI MALAYSIA SABAH

LIST OF FIGURES

Figure 1.1	Aibo, the robotic pet invented by Sony.	1
Figure 1.2	ASIMO, the humanoid robot developed by Honda.	2
Figure 1.3	The robot teacher SAYA.	2
Figure 1.4	Artist's conception of rover on Mars.	3
Figure 1.5	Six-legged walking robot CR200.	3
Figure 1.6	Parrot AR Drone.	4
Figure 1.7	Image of iRobot Packbot.	4
Figure 1.8	iRobot Roomba vacuum cleaner.	5
Figure 1.9	Overview of the proposed methods for solving path planning problem.	16
Figure 1.10	List of iterative methods considered in this study.	17
Figure 2.1	An overview of the stationary iterative methods.	34
Figure 2.2	An overview of the nonstationary iterative methods.	35
Figure 2.3	Deliberative Sense-Plan-Act architecture.	38
Figure 2.4	Reactive Sense-Act architecture.	38
Figure 2.5	Hybrid "three layer" architecture.	39
Figure 2.6	Trajectory of the robot from start to goal point.	42
Figure 2.7	Four configuration spaces are relatively simple to navigate.	46
Figure 2.8	Two configuration spaces are more complex and difficult to navigate.	47
Figure 2.9	The self-developed robot simulator software, Robot 2D Simulator.	48
Figure 2.10	(a) The real Khepera robot. (b) Sensor topology of the Khepera robot.	49
Figure 2.11	Placement of sensors and motors for the POINTROBOT.	50
Figure 2.12	Computational nodes of the configuration space for (a) standard or full-sweep, (b) half-sweep and (c) quarter-sweep iteration, respectively.	52
Figure 3.1	The computational molecules of the 5L approximation for (a) full-, (b) half- and (c) quarter-sweep cases, respectively.	58

Figure 3.2	Portion of the computational grid for the 5L about the point (i,j) for (a) full-, (b) half- and (c) quarter-sweep cases, respectively.	59
Figure 3.3	The computational molecules of the 9L approximation for (a) full-, (b) half- and (c) quarter-sweep cases, respectively.	68
Figure 3.4	Portion of the computational grid for the 9L about the point (i,j) for (a) full-, (b) half- and (c) quarter-sweep cases, respectively.	69
Figure 3.5	FSSOR-5L method considers all nodes in the mesh points.	79
Figure 3.6	HSSOR-5L method considers only half of the total nodes in the mesh points.	80
Figure 3.7	QSSOR-5L considers only quarter of the total nodes in the mesh points.	81
Figure 3.8	Grid for implementation of the 4-EGSOR-5L method.	84
Figure 3.9	Grid for implementation of the 4-EDGSOR-5L method.	87
Figure 3.10	Groups of four black points for the 4-MEGSOR-5L method.	90
Figure 3.11	FSSOR-9L method considers all nodes in the mesh points.	95
Figure 3.12	HSSOR-9L method considers only half of the total nodes in the mesh points.	96
Figure 3.13	QSSOR-9L considers only quarter of the total nodes in the mesh points.	97
Figure 3.14	Groups of four points are calculated using 9L approximation.	99
Figure 3.15	Group of four points with decoupled pairs.	103
Figure 3.16	Groups of nine points. In each group, 4-MEGSOR-9L method considers the four black nodes only.	106
Figure 3.17	The GDS picks the next node location with the lowest potential from its eight neighbouring points.	109
Figure 3.18	The classical robot control.	110
Figure 3.19	The behaviour-based control system.	111
Figure 3.20	The core behaviours of the POINTROBOT.	112

Figure 3.21	The Avoid-Obstacle behaviour. (a) The POINTROBOT turns 45°. (b) The POINTROBOT turns 90°. (c) The POINTROBOT turns 90° when it encounters a corner. (d) The POINTROBOT turns 135° when it encounters a corner from diagonal position.	114
Figure 3.22	The Follow-Wall behaviour. (a) Follow the wall for a specified period of time. (b) The POINTROBOT switches to avoid-obstacle behaviour before continuing its follow-wall behaviour. (c) The POINTROBOT changes its direction and switches to find-slope behaviour. (d) The POINTROBOT turns 90° and switches to findslope behaviour.	115
Figure 3.23	The Keep-Forward behaviour. In (a) and (c), the POINTROBOT has two options, whereas in (b) and (d) only one option is available.	117
Figure 3.24	The find-slope behaviour. (a) The timer is stopped if the goal (0,10) is found. (b) The POINTROBOT moves away from the flat regions (yellow).	120
Figure 4.1	The generated paths for Case 1 using Family of Point SOR methods via 5L with GDS.	135
Figure 4.2	The generated paths for Case 2 using Family of Point SOR methods via 5L with GDS.	136
Figure 4.3	The generated paths for Case 3 using Family of Point SOR methods via 5L with GDS.	137
Figure 4.4	The generated paths for Case 4 using Family of Point SOR methods via 5L with GDS.	138
Figure 4.5	The generated paths for Case 5 using Family of Point SOR methods via 5L with GDS.	139
Figure 4.6	The generated paths for Case 6 using Family of Point SOR methods via 5L with GDS.	140
Figure 4.7	Performance graph in terms of number of iterations for Case 1 using Family of Point SOR methods via 5L with GDS.	145

Figure 4.8	Performance graph in terms of number of iterations for Case 2 using Family of Point SOR methods via 5L with GDS.	146
Figure 4.9	Performance graph in terms of number of iterations for Case 3 using Family of Point SOR methods via 5L with GDS.	147
Figure 4.10	Performance graph in terms of number of iterations for Case 4 using Family of Point SOR methods via 5L with GDS.	148
Figure 4.11	Performance graph in terms of CPU time (in seconds) for Case 1 using Family of Point SOR methods via 5L with GDS.	149
Figure 4.12	Performance graph in terms of CPU time (in seconds) for Case 2 using Family of Point SOR methods via 5L with GDS.	150
Figure 4.13	Performance graph in terms of CPU time (in seconds) for Case 3 using Family of Point SOR methods via 5L with GDS.	151
Figure 4.14	Performance graph in terms of CPU time (in seconds) for Case 4 using Family of Point SOR methods via 5L with GDS.	152
Figure 4.15	Samples of Laplacian potentials for Case 1 using GDS-FSSOR-5L method.	153
Figure 4.16	Samples of Laplacian potentials for Case 2 using GDS-FSSOR-5L method.	154
Figure 4.17	Samples of Laplacian potentials for Case 3 using GDS-FSSOR-5L method.	155
Figure 4.18	Samples of Laplacian potentials for Case 4 using GDS-FSSOR-5L method.	156
Figure 4.19	The generated paths for Case 1 using Family of Four Point-Block SOR methods via 5L with GDS.	159
Figure 4.20	The generated paths for Case 2 using Family of Four Point-Block SOR methods via 5L with GDS.	160
Figure 4.21	The generated paths for Case 3 using Family of Four Point-Block SOR methods via 5L with GDS.	161
Figure 4.22	The generated paths for Case 4 using Family of Four Point-Block SOR methods via 5L with GDS.	162

Figure 4.23	The generated paths for Case 5 using Family of Four Point-Block SOR methods via 5L with GDS.	163
Figure 4.24	The generated paths for Case 6 using Family of Four Point-Block SOR methods via 5L with GDS.	164
Figure 4.25	Performance graph in terms of number of iterations for Case 1 using Family of Four Point-Block SOR methods via 5L with GDS.	169
Figure 4.26	Performance graph in terms of number of iterations for Case 2 using Family of Four Point-Block SOR methods via 5L with GDS.	170
Figure 4.27	Performance graph in terms of number of iterations for Case 3 using Family of Four Point-Block SOR methods via 5L with GDS.	171
Figure 4.28	Performance graph in terms of number of iterations for Case 4 using Family of Four Point-Block SOR methods via 5L with GDS.	172
Figure 4.29	Performance graph in terms of CPU time (in seconds) for Case 1 using Family of Four Point-Block SOR methods via 5L with GDS.	173
Figure 4.30	Performance graph in terms of CPU time (in seconds) for Case 2 using Family of Four Point-Block SOR methods via 5L with GDS.	174
Figure 4.31	Performance graph in terms of CPU time (in seconds) for Case 3 using Family of Four Point-Block SOR methods via 5L with GDS.	175
Figure 4.32	Performance graph in terms of CPU time (in seconds) for Case 4 using Family of Four Point-Block SOR methods via 5L with GDS.	176
Figure 4.33	The generated paths for Case 1 using Family of Point SOR methods via 9L with GDS.	179
Figure 4.34	The generated paths for Case 2 using Family of Point SOR methods via 9L with GDS.	180
Figure 4.35	The generated paths for Case 3 using Family of Point SOR methods via 9L with GDS.	181
Figure 4.36	The generated paths for Case 4 using Family of Point SOR methods via 9L with GDS.	182
Figure 4.37	The generated paths for Case 5 using Family of Point SOR methods via 9L with GDS.	183

Figure 4.38	The generated paths for Case 6 using Family of Point SOR methods via 9L with GDS.	184
Figure 4.39	Performance graph in terms of number of iterations for Case 1 using Family of Point SOR methods via 9L with GDS.	189
Figure 4.40	Performance graph in terms of number of iterations for Case 2 using Family of Point SOR methods via 9L with GDS.	190
Figure 4.41	Performance graph in terms of number of iterations for Case 3 using Family of Point SOR methods via 9L with GDS.	191
Figure 4.42	Performance graph in terms of number of iterations for Case 4 using Family of Point SOR methods via 9L with GDS.	192
Figure 4.43	Performance graph in terms of CPU time (in seconds) for Case 1 using Family of Point SOR methods via 9L with GDS.	193
Figure 4.44	Performance graph in terms of CPU time (in seconds) for Case 2 using Family of Point SOR methods via 9L with GDS.	194
Figure 4.45	Performance graph in terms of CPU time (in seconds) for Case 3 using Family of Point SOR methods via 9L with GDS.	195
Figure 4.46	Performance graph in terms of CPU time (in seconds) for Case 4 using Family of Point SOR methods via 9L with GDS.	196
Figure 4.47	The generated paths for Case 1 using Family of Four Point-Block SOR methods via 9L with GDS.	199
Figure 4.48	The generated paths for Case 2 using Family of Four Point-Block SOR methods via 9L with GDS.	200
Figure 4.49	The generated paths for Case 3 using Family of Four Point-Block SOR methods via 9L with GDS.	201
Figure 4.50	The generated paths for Case 4 using Family of Four Point-Block SOR methods via 9L with GDS.	202
Figure 4.51	The generated paths for Case 5 using Family of Four Point-Block SOR methods via 9L with GDS.	203
Figure 4.52	The generated paths for Case 6 using Family of Four Point-Block SOR methods via 9L with GDS.	204

Figure 4.53	Performance graph in terms of number of iterations for Case 1 using Family of Four Point-Block SOR methods via 9L with GDS.	209
Figure 4.54	Performance graph in terms of number of iterations for Case 2 using Family of Four Point-Block SOR methods via 9L with GDS.	210
Figure 4.55	Performance graph in terms of number of iterations for Case 3 using Family of Four Point-Block SOR methods via 9L with GDS.	211
Figure 4.56	Performance graph in terms of number of iterations for Case 4 using Family of Four Point-Block SOR methods via 9L with GDS.	212
Figure 4.57	Performance graph in terms of CPU time (in seconds) for Case 1 using Family of Four Point-Block SOR methods via 9L with GDS.	213
Figure 4.58	Performance graph in terms of CPU time (in seconds) for Case 2 using Family of Four Point-Block SOR methods via 9L with GDS.	214
Figure 4.59	Performance graph in terms of CPU time (in seconds) for Case 3 using Family of Four Point-Block SOR methods via 9L with GDS.	215
Figure 4.60	Performance graph in terms of CPU time (in seconds) for Case 4 using Family of Four Point-Block SOR methods via 9L with GDS.	216
Figure 4.61	Graph of reduction percentages in terms of number of iterations and CPU time for the proposed algorithms compared to the existing GDS-FSSOR-5L.	225
Figure 5.1	The generated paths for Case 1 using Family of Point SOR methods via 5L with LBBC.	229
Figure 5.2	The generated paths for Case 2 using Family of Point SOR methods via 5L with LBBC.	230
Figure 5.3	The generated paths for Case 3 using Family of Point SOR methods via 5L with LBBC.	231
Figure 5.4	The generated paths for Case 4 using Family of Point SOR methods via 5L with LBBC.	232
Figure 5.5	The generated paths for Case 5 using Family of Point SOR methods via 5L with LBBC.	233

Figure 5.6	The generated paths for Case 6 using Family of Point SOR methods via 5L with LBBC.	234
Figure 5.7	Performance graph in terms of number of iterations for Case 1 using Family of Point SOR methods via 5L with LBBC.	239
Figure 5.8	Performance graph in terms of number of iterations for Case 2 using Family of Point SOR methods via 5L with LBBC.	240
Figure 5.9	Performance graph in terms of number of iterations for Case 3 using Family of Point SOR methods via 5L with LBBC.	241
Figure 5.10	Performance graph in terms of number of iterations for Case 4 using Family of Point SOR methods via 5L with LBBC.	242
Figure 5.11	Performance graph in terms of number of iterations for Case 5 using Family of Point SOR methods via 5L with LBBC.	243
Figure 5.12	Performance graph in terms of number of iterations for Case 6 using Family of Point SOR methods via 5L with LBBC.	244
Figure 5.13	Performance graph in terms of CPU time (in seconds) for Case 1 using Family of Point SOR methods via 5L with LBBC.	245
Figure 5.14	Performance graph in terms of CPU time (in seconds) for Case 2 using Family of Point SOR methods via 5L with LBBC.	246
Figure 5.15	Performance graph in terms of CPU time (in seconds) for Case 3 using Family of Point SOR methods via 5L with LBBC.	247
Figure 5.16	Performance graph in terms of CPU time (in seconds) for Case 4 using Family of Point SOR methods via 5L with LBBC.	248
Figure 5.17	Performance graph in terms of CPU time (in seconds) for Case 5 using Family of Point SOR methods via 5L with LBBC.	249
Figure 5.18	Performance graph in terms of CPU time (in seconds) for Case 6 using Family of Point SOR methods via 5L with LBBC.	250
Figure 5.19	The generated paths for Case 1 using Family of Four Point-Block SOR methods via 5L with LBBC.	253
Figure 5.20	The generated paths for Case 2 using Family of Four Point-Block SOR methods via 5L with LBBC.	254

Figure 5.21	The generated paths for Case 3 using Family of Four Point-Block SOR methods via 5L with LBBC.	255
Figure 5.22	The generated paths for Case 4 using Family of Four Point-Block SOR methods via 5L with LBBC.	256
Figure 5.23	The generated paths for Case 5 using Family of Four Point-Block SOR methods via 5L with LBBC.	257
Figure 5.24	The generated paths for Case 6 using Family of Four Point-Block SOR methods via 5L with LBBC.	258
Figure 5.25	Performance graph in terms of number of iterations for Case 1 using Family of Four Point-Block SOR methods via 5L with LBBC.	263
Figure 5.26	Performance graph in terms of number of iterations for Case 2 using Family of Four Point-Block SOR methods via 5L with LBBC.	264
Figure 5.27	Performance graph in terms of number of iterations for Case 3 using Family of Four Point-Block SOR methods via 5L with LBBC.	265
Figure 5.28	Performance graph in terms of number of iterations for Case 4 using Family of Four Point-Block SOR methods via 5L with LBBC.	266
Figure 5.29	Performance graph in terms of number of iterations for Case 5 using Family of Four Point-Block SOR methods via 5L with LBBC.	267
Figure 5.30	Performance graph in terms of number of iterations for Case 6 using Family of Four Point-Block SOR methods via 5L with LBBC.	268
Figure 5.31	Performance graph in terms of CPU time (in seconds) for Case 1 using Family of Four Point-Block SOR methods via 5L with LBBC.	269
Figure 5.32	Performance graph in terms of CPU time (in seconds) for Case 2 using Family of Four Point-Block SOR methods via 5L with LBBC.	270
Figure 5.33	Performance graph in terms of CPU time (in seconds) for Case 3 using Family of Four Point-Block SOR methods via 5L with LBBC.	271
Figure 5.34	Performance graph in terms of CPU time (in seconds) for Case 4 using Family of Four Point-Block SOR methods via 5L with LBBC.	272
Figure 5.35	Performance graph in terms of CPU time (in seconds) for Case 5 using Family of Four Point-Block SOR methods via 5L with LBBC.	273