Far East Journal of Applied Mathematics
Volume 69, Number 2, 2012, Pages 111-123
Published Online: October 2012
Available online at http://pphmj.com/journals/fjam.htm
Published by Pushpa Publishing House, Allahabad, INDIA

IMPROVING THE STABILITY OF ALGORITHMS FOR PATH PLANNING BASED ON BOUNDARY VALUE PROBLEMS

Marcelo O. Silva[1], **Roseli A. F. Romero**[1], **Suely P. Oliveira**[2] **and David E. Stewart**[3]

[1]Departamento de Ciências da Computação, ICMC, USP
13560-970, São Carlos, SP - Caixa Postal 668, Brazil
e-mail: msilva@icmc.usp.br
rafrance@icmc.usp.br

[2]Department of Computer Science
The University of Iowa
Iowa City, IA 52242-1419, U. S. A.
e-mail: oliveira@cs.uiowa.edu

[3]Department of Mathematics
The University of Iowa
Iowa City, IA 52252, U. S. A.
e-mail: dstewart@math.uiowa.edu

Abstract

The purpose of this work is to develop algorithms for controlling multiple autonomous mobile robots, in dynamic environments. In such environments, the robots can do several complex tasks. One of the main tasks, for example, is to move to another region of the

2010 Mathematics Subject Classification: 93C85.

Keywords and phrases: upwind scheme, path planning, path planning BVP based, numerical methods, robotics.

Received June 8, 2012

environment without colliding with obstacles, which can be static or dynamic.

In general, this task has been done with path planning algorithms. In a dynamic environment, such as in versions of soccer for robots [2], it is crucial that the robot makes decisions in a short time. Therefore, the path planning algorithms cannot spend much run-time, because they can reduce the efficiency of the robot operations.

Locally Oriented Potential Fields (LOPF), proposed by [6], is a path planning technique that is applicable to multiple robots. This is based on numerical solutions of Boundary Value Problems which generate a given trajectory of a particular Elliptic Partial Differential Equation. It will be shown that this method is able to find a path (if it exists) from actual robot position to the goal, and can control multiple autonomous mobile robots, each one with distinct behaviors, by using the same environment grid.

1. Introduction

Robots have been used for industrial production processes automation. Such robots do not need to deal with dynamic environments (and unforeseen situations), and instead just perform well defined repetitive tasks. So, it was possible to program all the possible actions of the robotic agent for it to perform satisfactorily.

However, with technological developments, robots have begun to be used for other purposes in dynamics environments, as with entertainment (games), medicine (surgery), and performing tasks in dangerous or inaccessible environments to human (space and underwater). In this context, autonomous mobile robots are under development for a great number of complex tasks, which their predecessors (industrial robots) could not accomplish.

The main limitation on work with these robots is on how to control them, i.e., how to develop algorithms able to operate these complex machines. Complexity grows even more when takes in account, not just one robot but, a robot team. Then techniques are required to enable them to interact (effectively) with their environment. An essential part of this interaction is

the path planning algorithm utilized, that allows each robot to know their own way (based on their behavior) into environment.

Oriented Potential Fields (proposed by [10]) (equation (1)) and Locally Oriented Potential Fields (proposed by [6]) (equation (2)) are path planning techniques, in which the second (equation (2)) is applicable to control of multiple robots. These are based on numerical solutions of Boundary Value Problems (a BVP generates a given trajectory) of a particular Elliptic Partial Differential Equation [5]. As shown in [11], these techniques have a limitation in crucial part of PDE, that, in fact, tunes the behavior that robots will have

$$\nabla^2 P + \varepsilon\langle \nabla P,\ \mathbf{v}\rangle = 0 \tag{1}$$

with $\varepsilon \in \mathbb{R}$, $\mathbf{v} \in \mathbb{R}^2$ and $\| \mathbf{v} \| := \sqrt{v_x^2 + v_y^2} = 1$. Alternatively, we can make ε and $\mathbf{v}$ position-dependent:

$$\frac{\partial^2}{\partial x^2} P(x,\ y) + \frac{\partial^2}{\partial y^2} P(x,\ y) + \varepsilon(x,\ y)\left(v_x(x,\ y)\frac{\partial}{\partial x} P(x,\ y) + v_y(x,\ y)\frac{\partial}{\partial y} P(x,\ y)\right) = 0, \tag{2}$$

where $P, \varepsilon : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ and $\mathbf{v} : \Omega \to \mathbb{R}^2$ with $\mathbf{v}(x,\ y) = (v_x(x,\ y),\ v_y(x,\ y))$, and $v_x(x,\ y)^2 + v_y(x,\ y)^2 = 1$ for all $(x,\ y)$.

In this work, an upwind scheme is proposed for discretization of equations (1) and (2). In this way, it becomes possible to create new behaviors (independently of the value adopted for ε) to robots. These behaviors will be evaluated and tested.

2. Discretization

In this section, the stability of the Jacobi-Richardson and Gauss-Seidel numerical methods will be investigated [3, 7-9] when applied to solving linear systems obtained from HPF, OPF and LOPF. Although the proof will

be done only for LOPF, it is valid for the methods HPF and OPF since they are particular cases of LOPF. When $\varepsilon(x, y) \equiv 0$ and $\mathbf{v}(x, y) \equiv 0$, one obtains HPF and when $\varepsilon(x, y) = c \in \mathbb{R}$ and $\mathbf{v}(x, y) = \mathbf{w} \in \mathbb{R}^2$, one obtains OPF. The grid on which we work consists of points (x_i, y_j), where $x_{i+1} = x_i + h$ and $y_{j+1} = y_j + h$ for all i and j. We use $\varepsilon_{i,j} \coloneqq \varepsilon(x_i, y_j)$, $v_{x;i,j} \coloneqq v_x(x_i, y_j)$ and $v_{y;i,j} \coloneqq v_y(x_i, y_j)$. The discretized system of equations is for the unknowns $p_{i,j}$ which are approximations to the true solution: $p_{i,j} \approx P(x_i, y_j)$.

2.1. Central differences

Applying the finite difference method with central differences in equation (2) for a discrete domain represented by a grid with equally spaced cells, one obtains equation (3):

$$\begin{cases} \dfrac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{\Delta x^2} + \dfrac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{\Delta y^2} \\ \quad + \dfrac{\varepsilon_{i,j}}{2}\left[\dfrac{p_{i+1,j} - p_{i-1,j}}{\Delta x} v_{x;i,j} + \dfrac{p_{i,j+1} - p_{i,j-1}}{\Delta y} v_{y;i,j}\right] = 0 \\ \qquad \text{if } (x_i, y_j) \text{ is in the influence area of robot } k, \text{ and} \\ \dfrac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{\Delta x^2} + \dfrac{p_{i+1,j} - 2p_{i,j} + p_{i-1,j}}{\Delta y^2} = 0 \text{ otherwise.} \end{cases} \tag{3}$$

We assume that the grid is composed by equally spaced cells, i.e., $\Delta x = \Delta y = \delta$ and defining the parameter $\lambda = \varepsilon\delta/2$, one obtains equation (4):

$$p_{i,j} = \begin{cases} \dfrac{1}{4}\left[(1 + \lambda v_{x;i,j})\, p_{i+1,j} + (1 - \lambda v_{x;i,j})\, p_{i-1,j}\right. \\ \quad \left. + (1 + \lambda v_{y;i,j})\, p_{i,j+1} + (1 - \lambda v_{y;i,j})\, p_{i,j-1}\right] \\ \qquad \text{if } (x_i, y_j) \text{ is in the influence area of robot } k, \text{ and} \\ \dfrac{p_{i,j+1} + p_{i,j-1} + p_{i+1,j} + p_{i-1,j}}{4} \text{ otherwise.} \end{cases} \tag{4}$$

2.2. Upwind differences

The idea of upwinding is to use the values of P that are upwind from (x_i, y_j) for updating p_{ij}, as values of P at downwind from (x_i, y_j) will not have much effect on p_{ij} [1, p. 162-163]. For a general case, one obtains:

$$v_x(x, y)\frac{\partial}{\partial x}P \approx |v_{x;i,j}|\frac{p_{i+\mathrm{sgn}(v_{x;i,j}),j} - p_{i,j}}{h},$$

$$v_y(x, y)\frac{\partial}{\partial y}P \approx |v_{y;i,j}|\frac{p_{i,j+\mathrm{sgn}(v_{y;i,j})} - p_{i,j}}{h}, \qquad (5)$$

where

$$\mathrm{sgn}(x) = \begin{cases} +1, & x > 0, \\ 0, & x = 0, \\ -1, & x < 0. \end{cases}$$

Adding these approximations for $v_x\partial P/\partial x$ and $v_y\partial P/\partial y$ to the center difference approximation for $\nabla^2 P$ gives the equation

$$\frac{p_{i,j+1} - 2p_{i,j} + p_{i,j-1}}{h^2} + \frac{p_{i+i,j} - 2p_{i,j} + p_{i-1,j}}{h^2} + \varepsilon_{i,j}|v_{x;i,j}|\frac{p_{i+\mathrm{sgn}(v_{x;i,j}),j} - p_{i,j}}{h} + \varepsilon_{i,j}|v_{y;i,j}|\frac{p_{i,j+\mathrm{sgn}(v_{y;i,j})} - p_{i,j}}{h} = 0. \qquad (6)$$

Solving for p_{ij} in terms of $p_{i\pm1,j}$ and $p_{i,j\pm1}$ gives

$$p_{ij} = \frac{\left\{\begin{array}{l} p_{i-\mathrm{sgn}(v_{x;i,j}),j} + p_{i,j-\mathrm{sgn}(v_{y;i,j})} + (1 + h\varepsilon_{i,j}|v_{x;i,j}|)\,p_{i+\mathrm{sgn}(v_{x;i,j}),j} \\ + (1 + h\varepsilon_{i,j}|v_{y;i,j}|)\,p_{i,j+\mathrm{sgn}(v_{y;i,j})} \end{array}\right\}}{4 + h\varepsilon_{i,j}(|v_{x;i,j}| + |v_{y;i,j}|)}. \qquad (7)$$

3. Convergence Analysis

The following propositions are necessary for obtaining the numerical solution.

Proposition 1. *Let* $P : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ *be as in equation* (2), *in which* Ω *is discretized and path connected and that* $|\varepsilon(x, y)| < 2/h$. *Applying the finite difference method with central differences for discretization of equation* (2), *the Gauss-Seidel and Jacobi-Richardson methods converge when applied to resulting linear system.*

Proposition 2. *Let* $P : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ *be as in equation* (2), *in which* Ω *is discretized and path connected. Applying the finite difference method with central differences, by using upwind scheme for discretization of equation* (2), *the Gauss-Seidel and Jacobi-Richardson methods converge when applied to resulting linear system.*

Before we demonstrate the proof of Proposition 1, guaranteeing the convergence of LOPF, some definitions and theorems are presented because they are essential for understanding the proof.

Definition 3. A graph is *strongly connected* if and only if for any pair of nodes P_i, P_j there is a path between them

$$\overrightarrow{P_i P_{i_1}}, \overrightarrow{P_i P_{i_2}}, ..., \overrightarrow{P_{i_m} P_j}$$

connecting P_i and P_j [9, 4].

Definition 4. A matrix $A \in \mathcal{L}(\mathbb{R}^n)$ (where $\mathcal{L}(\mathbb{R}^n)$ is the space constituted by linear operators from $\mathbb{R}^n$ to $\mathbb{R}^n$) is *diagonally dominant* if

$$|a_{ii}| \geq \sum_{\substack{j=0 \\ j \neq i}}^{n} |a_{ij}|, \quad i = 1, ..., n \tag{8}$$

a strictly diagonally dominant if strict inequality holds in equation (8) for all

i. A matrix is *irreducible and dominant diagonally* if it is irreducible and dominant diagonally and at least for some index *i* in equation (8) the inequality is strict [9].

Definition 5. Given a matrix $A_{n\times n}$, there is an associated graph: Let *n* be distinct points: the vertices are points P_1, ..., P_n, and for each element non zero a_{ij} of *A* there is an edge from P_i to P_j. As a result, one obtains a digraph (directed graph) associated to matrix *A* [9]. If *A* is symmetric, then the graph associated with *A* is undirected.

Theorem 6. *The matrix* $A \in \mathcal{L}(\mathbb{R}^n)$ *is irreducible if and only if its associated graph is strongly connected* [9].

Theorem 7. *Let* $b \in \mathbb{R}^n$ *be arbitrary and* $A \in \mathcal{L}(\mathbb{R}^n)$ *be either strictly diagonally dominant or dominant diagonally and irreducible. Then the Jacobi-Richardson and Gauss-Seidel methods converge to* $A^{-1}b$ *for some* x_0 [9].

Now, based on the presented definitions, Proposition 1 can be proved.

Proof of Proposition 1. From equation (3), we have

$$\begin{cases} \dfrac{p_{i,\,j+1} - 2p_{i,\,j} + p_{i,\,j-1}}{\Delta x^2} + \dfrac{p_{i+1,\,j} - 2p_{i,\,j} + p_{i-1,\,j}}{\Delta y^2} \\ + \dfrac{\varepsilon}{2}\left[\dfrac{p_{i+1,\,j} - p_{i-1,\,j}}{\Delta x} v_x + \dfrac{p_{i,\,j+1} - p_{i,\,j-1}}{\Delta y} v_y \right] = 0, \end{cases}$$

taking $v_x(x,\,y) = v_y(x,\,y) = 0$ if $(x,\,y)$ is not in the region of influence of another robot. We also have $\Delta x = \Delta y = h > 0$. The coefficient of $p_{i,\,j}$ is then $-4h^{-2}$, while the coefficients of $p_{i\pm1,\,j}$ are $h^{-2} \pm \frac{1}{2}\varepsilon_{ij} v_{x;\,i,\,j} h^{-1}$ $= h^{-2}\left(1 \pm \frac{1}{2} h\varepsilon_{i,\,j} v_{x;\,i,\,j}\right)$ and the coefficients of $p_{i,\,j\pm1}$ are $h^{-2}\left(1 \pm \frac{1}{2} h\varepsilon_{i,\,j} v_{y;\,i,\,j}\right)$. Since $\|\mathbf{v}(x,\,y)\| \le 1$, and $\varepsilon(x,\,y) < 2/h$, for all

(x, y), the coefficients of $p_{i\pm1,\,j}$ and $p_{i,\,j\pm1}$ are all non-negative. Then summing the absolute values of the coefficients of $p_{i\pm1,\,j}$ and $p_{i,\,j\pm1}$ is the same as the sum without absolute values, which is $4h^{-2}$. This is the negative of the coefficient of $p_{i,\,j}$ in equation (3). If one or more of the points $(x_{i\pm1},\, y_j)$ and $(x_i,\, y_{j\pm1})$ are outside the interior of Ω, this will remove some off-diagonal coefficients, and thus reduce the sum of off-diagonal entries for the equation for $p_{i,\,j}$. Thus, the inequality equation (8) is satisfied for all rows of the linear system for the $p_{i,\,j}$'s with strict inequality if $(x_i,\, y_j) \in \Omega$ but one of $(x_{i\pm1},\, y_j)$ or $(x_i,\, y_{j\pm1})$ is not in the interior of Ω.

Therefore, Theorem 7 guarantees the convergence of Gauss-Seidel and Jacobi-Richardson methods when applied to equation (3). □

Using upwind differences, however, removes the restriction that $\varepsilon(x,\, y) < 2/h$.

Proposition 8. *Let* $P : \Omega \subseteq \mathbb{R}^2 \to \mathbb{R}$ *be as in equation* (2), *in which* Ω *is discretized and path connected. Applying the finite difference method with upwind differences for discretization of equation* (2), *the Gauss-Seidel and Jacobi-Richardson methods converge when applied to resulting linear system.*

Proof. The proof of this starts with equation (6):

$$\frac{p_{i,\,j+1} - 2p_{i,\,j} + p_{i,\,j-1}}{h^2} + \frac{p_{i+i,\,j} - 2p_{i,\,j} + p_{i-1,\,j}}{h^2} + \varepsilon_{i,\,j}|v_{x;i,\,j}|\frac{p_{i+\mathrm{sgn}(v_{x;i,\,j}),\,j} - p_{i,\,j}}{h} + \varepsilon_{i,\,j}|v_{y;i,\,j}|\frac{p_{i,\,j+\mathrm{sgn}(v_{y;i,\,j})} - p_{i,\,j}}{h} = 0.$$

Note that all the coefficients of $p_{i\pm1,\,j}$ and $p_{i,\,j\pm1}$ are non-negative and the

coefficient of $p_{i,j}$ is always negative. Assuming that $(x_i, y_j) \in \Omega$ and $(x_{i\pm1}, y_j), (x_i, y_{j\pm1}) \in \Omega$, the sum of the off-diagonal coefficients in the above equation is $4h^{-2} + \varepsilon_{i,j}(|v_{x;i,j}| + |v_{y;i,j}|)h^{-1}$, while the absolute value of the coefficient of $p_{i,j}$ is also $4h^{-2} + \varepsilon_{i,j}(|v_{x;i,j}| + |v_{y;i,j}|)h^{-1}$. Thus equation (8) is satisfied in this case. If one of $(x_{i\pm1}, y_j), (x_i, y_{j\pm1})$ $\notin$ interior Ω, then this reduces the sum of the off-diagonal coefficients, and so equation (8) becomes a strict inequality. Then Theorem 7 guarantees the convergence of Gauss-Seidel and Jacobi-Richardson methods. □

4. Experiments

In Figure 1, the convergence of the CPO equations with Gauss-Seidel, both using sequential implementation and a parallel implementation in CUDA is shown using centered versus upwind differences for various constant values of $\varepsilon (h = 1)$. The grid used as 85×65, as this is suitable for the robot soccer challenge.

It can be seen that all the methods converge for $\varepsilon < 2$, as was shown theoretically through the results of the previous section. In this experiment, only when $\varepsilon > 3.4$ do we see divergence for the centered difference discretization, while the upwind version does not have any problems with convergence.

Tables 1 and 4 show the execution times (in milliseconds) of the solution methods for the boundary value problems for LOPF (Locally Oriented Potential Fields) using central differences and upwinding, respectively. Several different iterative methods were compared: JR (Jacobi-Richardson), WJR (Weighted Jacobi-Richardson), GS (Gauss-Seidel) and SOR (Successive Over-Relaxation) [3, 9, 12]. For SOR, optimal over-relaxation parameters ω were computed (ω_{opt} was in the range 1.8 to 1.9). Taking the time for sequential Gauss-Seidel as a base (29.19 ms), the speedups were calculated in Tables 2 and 5. The number of iterations are shown in Tables 3

and 6. It can be seen that upwind differences are the fastest for both the total execution time, and the number of iterations per millisecond.

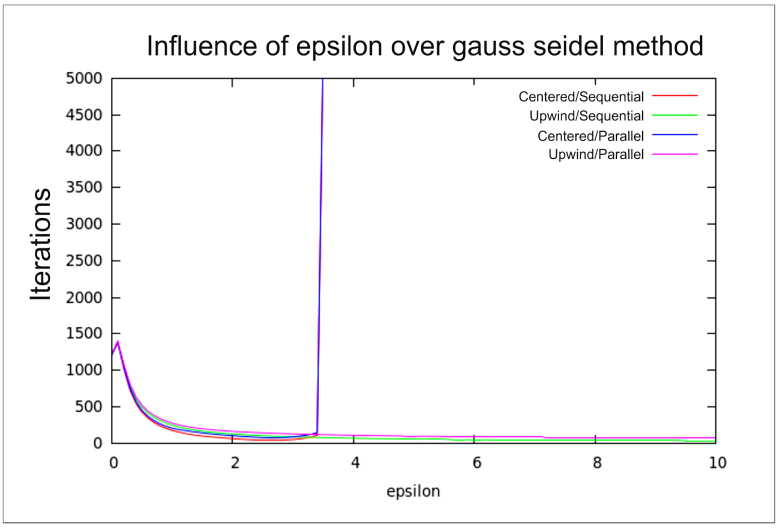

Figure 1. Gauss Seidel-ε.

Table 1. Execution time-centered differences

	Sequential/CPU		Parallel/GPU	
	Average	Range	Average	Range
JR	-	-	0.88	[0.73, 1.02]
WJR	-	-	0.75	[0.63, 0.86]
GS	29.18	[22.89, 35.48]	1.16	[0.99, 1.34]
SOR	21.30	[18.88, 23.72]	0.70	[0.64, 0.77]

Table 2. Speedup-centered differences

	Sequential/CPU	Parallel/GPU
JR	-	33.16
WJR	-	38.91
GS	1	25.16
SOR	1.37	41.69

Table 3. Number of iterations-centered differences

	Sequential/CPU		Parallel/GPU	
	Average	Range	Average	Range
JR	-	-	49.13	[40.50, 57.76]
WJR	-	-	40.78	[33.85, 47.70]
GS	20.43	[22.48, 34.38]	32.46	[27.31, 37.60]
SOR	20.52	[18.20, 22.83]	18.32	[16.82, 20.84]

Table 4. Execution time-upwind differences

	Sequential/CPU		Parallel/GPU	
	Average	Range	Average	Range
JR	-	-	0.83	[0.67, 0.98]
WJR	-	-	0.68	[0.57, 0.80]
GS	28.41	[22.41, 34.39]	1.07	[0.90, 1.25]
SOR	16.92	[14.50, 19.35]	0.85	[0.76, 0.93]

Table 5. Speedup-upwind differences

	Sequential/CPU	Parallel/GPU
JR	-	35.17
WJR	-	42.91
GS	1.03	27.27
SOR	1.73	34.33

Table 6. Number of iterations-upwind differences

	Sequential/CPU		Parallel/GPU	
	Average	Range	Average	Range
JR	-	-	49.13	[40.50, 47.76]
WJR	-	-	40.78	[33.85, 47.70]
GS	28.44	[22.54, 34.37]	27.42	[22.72, 37.12]
SOR	17.16	[14.72, 19.61]	21.30	[19.07, 23.52]

5. Conclusions

Following the results presented in the previous section, it can be seen that the use of the GPU strongly reduces the time for the execution of the method (with up to 40x the speed of the CPU), in spite of the transfer times between CPU and GPU. In the current implementation, the error calculation is performed on the CPU rather than GPU; future versions of the code will solve this problem. Furthermore, by using upwind differencing, we can treat problems with $\varepsilon(x, y) > 2/h$ without difficulty.

The number of iterations remains essentially the same between the CPU and GPU versions. This indicates that the execution of the methods can be greatly sped up using massive parallelism of the type supplied by the GPU and CUDA architecture. No individual method is better than all the others, but it can be said that the SOR/CUDA and WJR/CUDA are faster and more efficient than the others. There is little to distinguish between the SOR/CUDA and WJR/CUDA methods.

The execution time using upwind differences with WJR/CUDA is only 0.68 milliseconds as shown in Table 4 allows over 1,400 executions per second. This is more than sufficient for real time execution, even for path planning for rapidly moving situations like robot soccer

References

[1] Uri M. Ascher, Numerical methods for evolutionary differential equations, Computational Science and Engineering, Vol. 5, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2008.

[2] Jacky Baltes, N. Michael Mayer, John Anderson, Kuo-Yang Tu and Alan Liu, The humanoid leagues in robot soccer competitions, Proceedings of the IJCAI Workshop on Competitions in Artificial Intelligence and Robotics, 2009, pp. 9-16.

[3] R. L. Burden, J. D. Faires and A. C. Reynolds, Numerical Analysis, PWS Publ. Co., 1978.

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, Introduction to Algorithms, 2nd ed., McGraw-Hill, 2001.

[5] R. Courant and D. Hilbert, Methods of Mathematical Physics, Vol. 2, John Wiley and Sons, New York, 1962.

[6] G. Faria, R. A. F. Romero and E. Prestes, Multi-robot control by using boundary value problems, 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Beijing, Proceedings, 2006.

[7] B. Neide Franco, Cálculo Numérico, Pearson Prentice-Hall, 2006.

[8] Suely Oliveira and David Stewart, Writing Scientific Software: A Guide to Good Style, Cambridge University Press, Cambridge, UK, 2006.

[9] James M. Ortega, Numerical Analysis - A Second Course, Academic Press, 1972.

[10] E. Prestes e Silva, Jr., Marco A. P. Idiart, Marcelo Trevisan and Paulo M. Engel, Autonomous learning architecture for environmental mapping, J. Intelligent and Robotic Systems 39 (2004), 243-263.

[11] M. O. Silva, R. A. F. Romero and G. A. P. Caurin, Analysis of locally oriented potential fields, N. Lau R. Bianchi, L. P. Reis and L. Correia, eds., Proceedings of IROBOT 2008-3rd International Workshop on Intelligent Robotics, October 14th, Lisbon University Institute-ISCTE, Lisbon, Portugal, 2008.

[12] G. D. Smith, Numerical Solution of Partial Differential Equations: Finite Difference Methods, Oxford University, 1992.