The 2010 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

A Real-Time Path Planner for a Smart Wheelchair Using Harmonic Potentials and a Rubber Band Model

R. Hong* and G. N. DeSouza**

Abstract— **We present an efficient path planner for smart wheelchairs based on harmonic potential fields. While the use of harmonic fields can always guarantee finding an existing path, they are extremely computational intensive and a sufficiently detailed map of the environment may lead to an unfeasible solution for the path. Also, since our target application is for the navigation of a smart wheelchair, for people with severe disabilities, the path provided by the harmonic field is frequently too sharp and needs to be smoothened. In order to address the first problem, we propose a parallel algorithm implemented using Graphics Processor Units (GPUs) on the Compute Unified Device Architecture (CUDA) platform. And for the second problem, we developed a rubber band model that provides extra forces to be added to the attracting forces of the harmonic fields. This model assumes that the path is an elastic line, a rubber band, connecting the source and destination points. This rubber band simulates the internal tension forces trying to tighten the line. As the result section demonstrates, both the original path from the harmonic field alone and the path smoothened by the rubber band model have approximate the same length, but the first path contains many *bumps, sharp angles,* and *zig-zags*, while the second one provides a much more comfortable ride for the passenger of the wheelchair. Either one is executed in real-time, allowing the proposed method to be used for real navigation of smart wheelchairs.**

I. Introduction

Path planning is a crucial task for any mobile robot navigation [1]. Usually, we can find two approaches for this problem: 1) using topological graphs; and 2) using grid-based maps; and while other methods, such as in [2] and [3], may not fall directly into this dichotomy, they are ultimately a hybrid combination of both types ([2]) or a hierarchical structuring from a coarse scale (topological) all the way to a finer one (grid).

In the case of grid-based maps, a common method will involve the calculation of either: fluid dynamics models [4]; snake models [5]; elastic band models [6][7]; or potential fields [8], [9], and more effectively, harmonic potential fields [10], [11], [12], which provide the most robust way to generate paths. That is, while potential fields and other models may be easier to calculate, they may also get stuck in local minima. However, harmonic potentials can provide a guarantee that a path will always be found - i.e. if one exists.

Once again, the problem in using harmonic fields is that it requires repeated updates of the potential values at every cell of the grid. These updates are in turn a function of the potential of the neighboring cells, which leads to a recursive and quite time-consuming algorithm. One simplification of this recursion that usually translates into an algorithm speedup is achieved by performing a sequential row-wise update - starting from the upper-left corner of the grid [10] and moving down to the bottom-right corner. Another form of speedup is obtained by implementing the harmonic fields using a closed-form and analytic solution, but this presents a serious drawback since the number of primitives that must be created to represent obstacles becomes limited [13]. Despite the method used for speed up, the gains are usually not justified given the loss of accuracy and the per-cell basis, and therefore still time-consuming nature of any possible solution.

As we know, a more effective way to speed up time consuming algorithms is through the use of parallel computation. Even more pertinent to our problem, since a fine and detailed grid of the environment may require millions of cells [14], we propose the parallelization through the use of General Purpose Graphics Processor Units (or GPGPUs). The use of GPGPUs together with the use of CUDA [15] indeed translate into an easy and standard platform for the implementation of harmonic fields.

After generating an initial path, it is common to optimize such path in order, for example, to smoothen it [16]. Some authors proposed the idea of neural networks, generic algorithm [17], or even splines [18] in order to smoothen the path. Our motivation to optimize the path comes from the use of the algorithm to navigate a smart wheelchair. In that case, an optimized solution must include: efficiency in avoiding static as well as dynamic obstacles; smoothness of the path; and total length of the path [19][20]. In the proposed method, the optimization comes from a rubber band model that regards the path as an elastic line [21]. This elastic path can reduce unnecessary curves because of its internal tension forces, which tends to pull the elastic back to a straight position.

In a nutshell, we assume that the path is made of small segments and for each segment, the harmonic potential forces compete with the tension forces from neighboring segments, pulling the path at the same time away from obstacles and along its tangential direction. By doing so, the segments will move along the direction of summation of the force and will stop moving when these forces reach a balance.

II. Path Planning and Harmonic Potential Fields

In grid-based maps, the idea is to represent the environment as a 2D grid. The grid is basically a ground plane projection of any object detected by the robot, in our

Department of Electrical and Computer Engineering, University of Missouri, 349 Eng. Building West, Columbia, MO, USA
*rhc42@mizzou.edu ** DeSouzaG@missouri.edu

case using the laser range sensor. When potential fields are applied on top of such grids, obstacles are described by high potentials or *hills* that must be avoided, and source and destination points are the zero potentials or *valleys*. The path towards the destinations is defined along the valleys. Unfortunately, due to interaction between multiple objects, valleys are not unique in potential fields. On the other hand, these same problem of local minima disappear when we use harmonic potential fields [10], [12].

A. Harmonic Function

A harmonic function on a domain $\Omega \subset R^n$ is a function which satisfies Laplace's equation. That is:

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

This same function can be discretized and the numerical solution of Laplace's equation becomes ([10]):

$$\begin{aligned} u^{(k+1)}(x,y) = \tfrac{1}{4}[(u^{(k)}(x+1,y) + u^{(k)}(x-1,y) \\ + \quad u^{(k)}(x,y+1) + u^{(k)}(x,y-1)] \end{aligned} \quad (1)$$

where $u(x,y)$ represents the discrete sample of ϕ at coordinates (x,y) of the R^2 grid, and k is the iteration number. That is, at each iteration, a grid cell of ϕ is updated with the average value of its neighbors. On a sequential computer, this solution is usually implemented as follows:

$$\begin{aligned} u^{(k+1)}(x,y) \quad = \quad \tfrac{1}{4}[u^{(k)}(x+1,y) + u^{(k+1)}(x-1,y) \\ + \quad u^{(k)}(x,y+1) + u^{(k+1)}(x,y-1)] \end{aligned}$$

That is, the next values of the top and left neighbors of the current cell are updated and used in the calculation of that same cell. This speed up of the algorithm allows for the values of next iteration to quickly propagate through the grid. However, it also distorts the real value of the harmonic potentials.

B. Definitions

In order to explain the proposed method, a few basic elements need to be defined [17]. A *goal* is a grid cell with the lowest harmonic value ($u^{(k)}(x,y) = 0$). This value is fixed and it will never be affected by its neighbor's values. An *obstacle* is any cluster of cells blocking a potential path towards the goal. Its value is maximum ($u^{(k)}(x,y) = 1$) and is also never affected by its neighbors. However, its position may change as the environment is dynamic. *Free space* is any grid cell that does not contain an obstacle or the goal. The value of the harmonic potential in the free space is initialized with the median range and it is updated at each iteration.

A path to the goal is given by an index matrix, $Idx(x,y)$, which for every position (x,y) contains the index of the neighbor with the lowest harmonic potential. That is, $Idx(x,y) = min[u^{(k)}(x+1,y), u^{(k+1)}(x-1,y), u^{(k)}(x,y+1), u^{(k+1)}(x,y-1)]$.

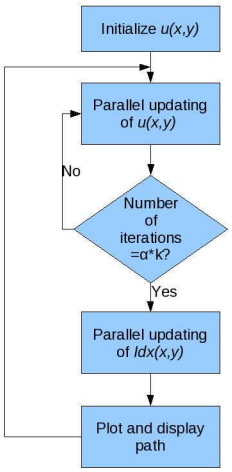

Fig. 1. Flowchart of the algorithm for calculation of the harmonic potential fields

III. Parallel Implementation

Due to the limitation on the number of pages, in this section it will suffice to say that the above algorithm translates quite nicely into the parallel paradigm of GPU computing [15]. In that sense, a single CUDA program, namely a kernel function, can be implemented to carry out the calculation of each grid cell. This same kernel function is then multiply instantiated by the CUDA platform and the calculations of the various grid cells is performed in parallel by the GPU. All that is left to be done by the CUDA *host()* function is to perform the k iterations. A programmer in the CUDA environment must only be cautious not to perform too many CPU to/from GPU memory transfers as those can be very time consuming. One such transfer is required when the program needs to display the harmonic potentials for purpose of user interfacing. For that reason, our algorithm limits such display to a multiple of k iterations. Figure 1 depicts the complete algorithm. The first block of the flowchart is the initialization and involves transfer of data from CPU to GPU memory. The next three blocks of the flowchart are performed in parallel by the GPU and consist of the main processing of the harmonic potential fields. The last block is simply for display purposes.

IV. Rubber Band Model for Path Optimization

In this section we explain how the proposed rubber band model is employed to optimize the path obtained by the harmonic potential fields. This idea of a rubber band is not totally new, but its use ([22]) has been mostly to define obstacle contours. Here, we combine the ideal of rubber band

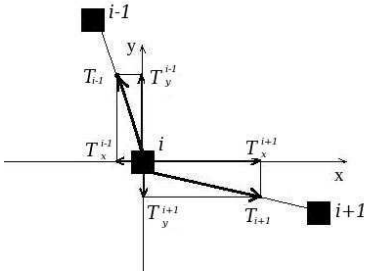

Fig. 2. Tensions exerted by consecutive cells along the path

model and harmonic potentials to define the path, which we regard as a linked list of grid cells. The two immediately adjacent cells in the link, i.e. the previous and the next cells along the link from the current cell, exert internal forces on that same cell. Figure 2 illustrates this idea for the cell i and its previous and next cells in the path, $i-1$ and $i+1$, respectively.

Every cell in the path is affected by two kinds of forces: the internal tension (rubber band) forces $\overrightarrow{T}_{i\pm1}$, and the potential force $\overrightarrow{F}$. The position of a cell in the path is given by the pair (x, y) that leads to the resultant forces to be minimum. That is:

$$(\hat{x},\hat{y}) = arg_{(x,y)}min(\left\|\overrightarrow{F} + \overrightarrow{T}_{i+1} + \overrightarrow{T}_{i-1}\right\|) \qquad (2)$$

$$P_i^{k+1} = P_i^k + \delta * (\hat{x},\hat{y})$$

where δ represents a small step ($0 < \delta < 1$) used to move the position of the current cell in the path at each iteration. That is, let us assume that the current coordinates of the i^{th} cell is (x_i, y_i), and the coordinates of the two neighbors are (x_{i-1}, y_{i-1}) and (x_{i+1}, y_{i+1}), respectively. The resultant of the forces on the cell i, as shown by Figure 3, provides the direction and intensity with which the cell should be moved in order for the forces to reach equilibrium. This direction and intensity are multiplied by δ so that the cell is moved only a fraction of that value at each iteration.

The last component of these calculations is the force $\overrightarrow{F}$ derived from the harmonic potential. This force is calculated using:

$$F = 1/(1 - u^k(\hat{x},\hat{y})) - 1/(1 - u^k(x_i, y_i))$$

where $u(x_i, y_i)$ represents the harmonic potential at the current position of the cell in the path given by eq. (1), and $u(\hat{x},\hat{y})$ represents the harmonic potential of the position to which the cell i is being dragged.

Figure 3 summarizes the complete idea of the harmonic potentials and the internal tension (elastic) forces of the model. In the figure, red blocks represent obstacles (walls)

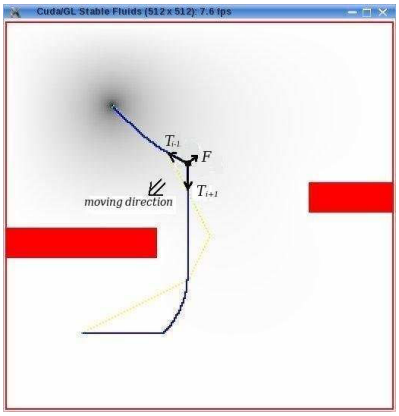

Fig. 3. Here, it shows the resultant forces on cell i, as well as the harmonic potential path (blue) being optimized by the rubber band model (green).

and the black dot is the desired destination of the smart wheelchair. The darker is the color in free space, the lower is the harmonic potential value. The figure also shows how the path obtained from the simple application of harmonic fields (blue path) compares to the one being optimized by the rubber band model (green path).

A. Parallel Computation of the Rubber Band Model

As before, the algorithm for optimizing the path using the rubber band model was also implemented under CUDA and executed in parallel by a GPU. For that, we also defined a single kernel function that was once again instantiated for each cell in the grid. This algorithm, which replaces the fourth block in the original flowchart in Figure 1, is presented in more detail in Figure 4.

V. Experimental Results

We conducted various tests of the algorithm for indoor navigation using a robot simulator for the Pioneer P3-DX (MobileSim). The actual control of the robot is carried out by setting its speed constant and providing constant adjustments for its heading according to the path calculated by the GPU module. That is, 1) the laser data from the simulator is used to define obstacles on the grid; 2) the GPU module calculates the path according to such obstacles and computes the turning angles necessary to follow that path; and 3) the commands with the turning angles are sent back to the simulator. The tests used four different scenarios, which are presented in Figure 5. The same figure depicts the paths resulting from the application of the harmonic potentials alone (left column) and the same paths optimized by the rubber band model (right column).

In order to measure the effect of the optimization on the lengths of the paths, we calculated such lengths with and without the optimization. Table I summarize these results.

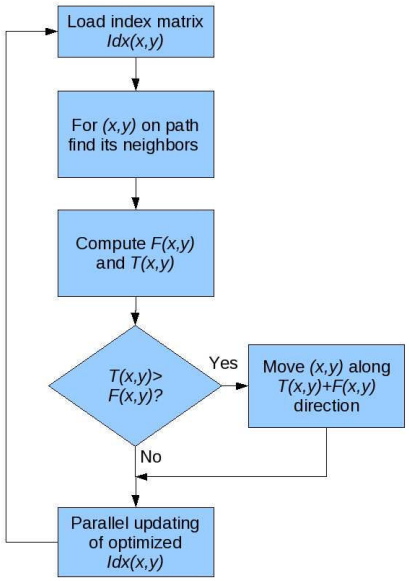

Fig. 4. Flowchart of the algorithm for optimization of the path obtained by the harmonic potentials.

Length (pixels)	Scene1	Scene2	Scene3	Scene4
Harmonic Path	2.34m	3.80m	2.85m	3.22m
Optimized Path	2.34m	3.72m	2.75m	3.26m

TABLE I

STATISTICS OF THE LENGTH FOR THE PATHS FOR THE HARMONIC POTENTIALS ONLY VERSUS THE OPTIMIZED PATH

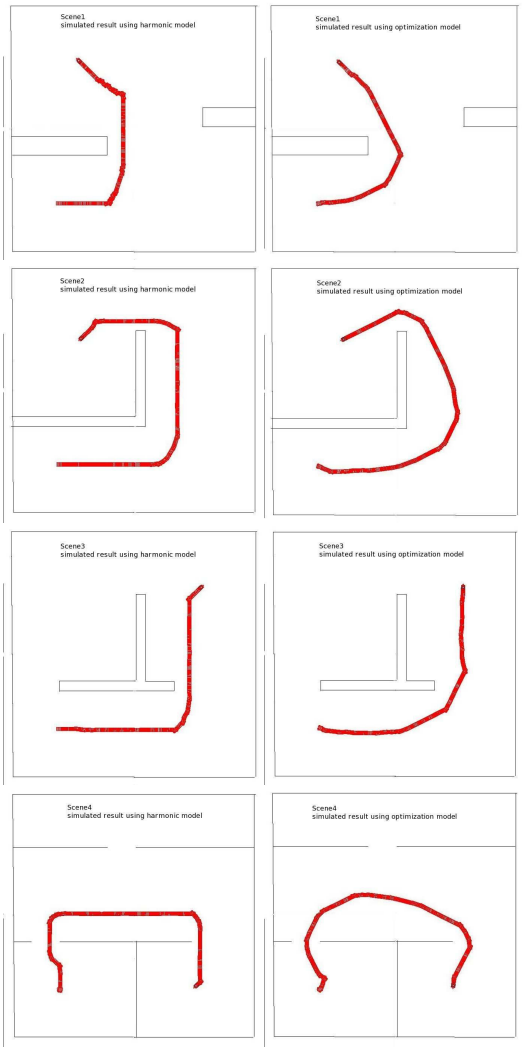

Fig. 5. Path obtained by the use of harmonic potentials alone (left column) and the optimized rubber band model (right column)

Finally, in order to appreciate how much smoother the optimized path is in comparison with the original path obtained by the harmonic potentials, we computed the histogram of turning angles used during the navigation following each of the two paths. As Figure 6 indicates, the turning angles of the optimized path concentrates on the small angular values, namely around 20^o, whereas the concentration of turning angles for the non-optimized path peaks at higher values and spreads up to 50^o, 60^o, or even 70^o. It should go without saying that a large turning angle makes it difficult for the wheelchair to follow the path. So, as it can be inferred from these histograms, the optimized path is a smoother and therefore more comfortable path for the passenger.

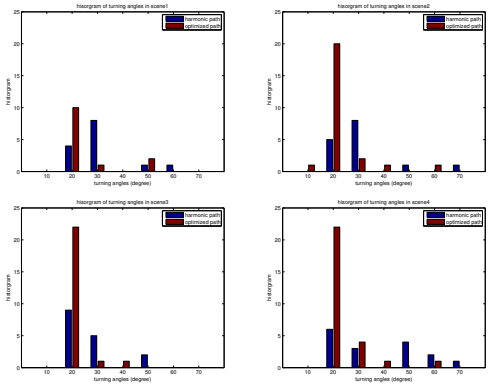

Fig. 6. Histograms of turning angles used during navigation of the path given by harmonic potentials and by the rubber band model

A. Dynamic Environments

In order to test our algorithm in terms of its real time performance, we also built a scenario where the mobile robot would have to avoid moving obstacles. For these scenarios, we programed MobileSim to simulate up to 10 robots at

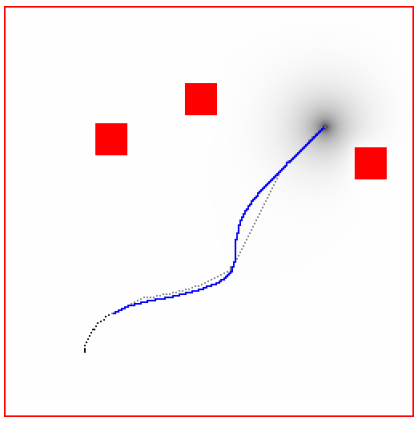

Fig. 7. Path-planning in Dynamic Environments

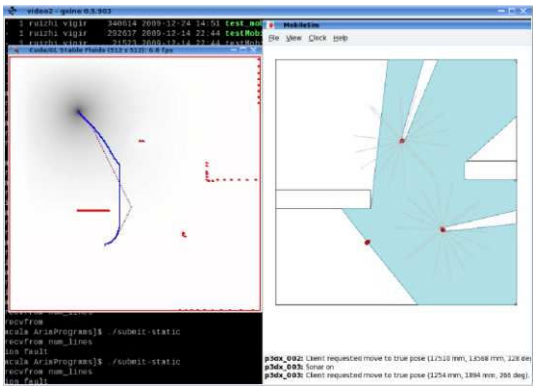

Fig. 8. Sample of a test scenario: on the left, the information sensed by the robot and calculated path; on the right, the simulator screen.

the same time. We then controlled one of these robots using our algorithm, while the other robots were controlled using random paths – that is, the other robots were programed to wander randomly inside the same environment. Finally, the main robot would use the laser range sensor to detect these moving obstacles, which were represented on the grid as squares centered at the detected location. Figure 7 presents the calculated path on the GPU for one such experiment. The tests consisted of recording the time to reach the destination, the length of the final path, and how many times the path had to be interrupted and recreated due to unavoidable obstacles. If the robot collided with any obstacle before reaching the destination, this would be considered as failed attempt and so recorded. This test was repeated 100 times and results recorded and averaged.

As shown in Figure 7, the mobile robot attempted to follow as much as possible the optimized path (blue line), but due to the dynamic nature of the environment, that was not always possible. The black dotted curve recorded the actual path followed by the robot. The reader should keep in mind that the figure displays simply a snapshot of the environment at a certain time t when the obstacles (red blocks) occupied the displayed positions. However, the same obstacles may have caused the path to shift, which cannot be easily seen by this snapshot. A mpeg video with the actual sequence of motions of the mobile robot as well as the obstacles is being submitted with this paper and will be available in the proceedings of the conference.

Without any obstacles, the mobile robot required 14 seconds to navigate the 2.21m of the path from source to destination. After introducing 3 moving obstacles, the mean time to reach the goal increased to 18.33 seconds; the mean length of the path went up to 2.73m; and the average number of times the path was interrupted and recreated was 0.42 times. With 3 moving obstacles, the mobile robot reached its destinations 100% of the time. After introducing 10 moving obstacles, the same numbers became: 27.08 seconds to complete, path length of 3.85m, average number of interruptions was 2.36 times, and in 87% of the trials the robot succeeded in reaching the destination.

Finally, 8 shows a snapshot of the computer screen with two windows: on the left side, we present the internal representation (GPU) of the sensed information used to calculate the path; and on the right side, we see the simulator screen, with the actual robots, the laser range sensor information, and the walls of the environment.

VI. Conclusions

We describe a path planning approach based on harmonic fields and optimized by a rubber band model. Both methods were successfully implemented in real time using a parallel algorithm. Both methods efficiently avoid obstacles and are capable of reaching the final goal, unless an large number of moving obstacles systematically block their path. The optimized path reduced drastically the sharp angles obtained with the original harmonic fields alone.

The major problem of the proposed algorithm was observed during the dynamic tests. In those cases, if a moving obstacle crosses the path of the robot, a new path is always calculated, even when the path is interrupted far ahead in front of the current position of the robot. This, some times, leads to the robot to move sideways or even backwards, when the best solution in those cases would obviously be just to halt the motion and await for the obstacle to move again out of the way. This problem will be approached in the future by incorporating a predictive model to the harmonic potentials. That is, instead of computing the harmonic potentials (high or low values) based on the current snapshot of the occupancy of the environment, we will incorporate an occupancy map that will reflect the state of the environment after a certain time window into the future.

References

[1] G. N.DeSouza and A. C.Kak, "Vision for mobile robot navigation:A survey," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 237–267, Feb. 2002.

[2] R. SIM AND J. J. LITTLE, "AUTONOMOUS VISION-BASED EXPLORATION AND MAPPING USING HYBRID MAPS AND RAO-BLACKWELLISED PARTICLE FILETERS," IN *International Conference on Intelligent Robots and Systems*, IEEE/RSJ, OCT. 2006.

[3] B. KUIPERS, J. MODAYIL, P. BEESON, M. MACMAHON, AND F. SAVELLI, "LOCAL METRICAL AND GLOBAL TOPOLOGICAL MAPS IN THE HYBRID SPATIAL SEMANTIC HIERARCHY," IN *IEEE Int. Conf. on Robotics & Automation*, PP. 4845–4851, 2004.

[4] D. KEYMEULEN AND J. DECUYPER, "THE FLUID DYNAMICS APPLIED TO MOBILE ROBOT MOTION: THE STREAM FIELD METHOD," IN *International Conference on Robotics and Automation*, 1994.

[5] Y. CHENG, P. JIANG, AND Y. F. HU, "A DISTRIBUTED SNAKE ALGORITHM FOR NOBILE ROBOTS PATH PLANNING WITH CURVATURE CONSTRAINTS," IN *International Conference on Systems, Man and Cybernetics*, 2008.

[6] S. QUINLAN AND O. KHATIB, "ELASTIC BANDS: CONNECTING PATH PLANNING AND CONTROL," IN *International Conference on Robotics and Automation*, 1993.

[7] T. SATTEL AND T. BRANDT, "GROUND VEHICLE GUIDANCE ALONG COLLISION-FREE TRAJECTORIES USING ELASTIC BANDS," IN *American Control Conference*, 2005.

[8] P. VADAKKEPAT, K. C. TAN, AND W. MING-LIANG, "EVOLUTIONARY ARTIVICIAL POTENTIAL FIELDS AND THEIR APPLICATION IN REAL TIME ROBOT PATH PLANNING," *Proceedings of the 2000 Congress on Evolutionary Computation*, VOL. 1, PP. 256–263, 2000.

[9] Y. K. HWANG AND N. AHUJA, "A POTENTIAL FIELD APPROACH TO PATH PLANNING," *IEEE Transactions on Robotics and Automation*, VOL. 8, NO. 1, PP. 23–32, 1992.

[10] C. I. CONNOLLY AND R. A. GRUPEN, "ON THE APPLICATIONS OF HARMONIC FUNCTIONS TO ROBOTICS," *Journal of Robotic Systems*, VOL. 10, NO. 7, PP. 931–946, 1993.

[11] A. AHMAND AND MASOUD, "A DISCRETE HARMONIC POTENTIAL FIELD FOR OPTIMUM POINT-TO-POINT ROUTING ON A WEIGHTED GRAPH," *International Conference on Intelligent Robots and Systems*, PP. 9–15, OCT. 2006.

[12] R. DAILY AND D. M. BEVLY, "HARMONIC POTENTIAL FIELD PPATH PLANNING FOR HIGH SPEED VEHICLES," IN *American Control Conference*, 2008.

[13] R. DAILY AND D. M. BEVLY, "HARMONIC POTENTIAL FIELD PATH PLANNING FOR HIGH SPEED VEHICLES," IN *American Control Conference*, 2008.

[14] H.-C. CHANG AND J.-S. LIU, "HIGH-QUALITY PATH PLANNING FOR AUTONOMOUS MOBILE ROBOTS WITH Î·3-SPLINES AND PARALLEL GENETIC ALGORITHMS," IN *Robotics and Biomimetics*, 2008.

[15] D. B. KIRK AND W. MEI W. HWU, *Programming Massively Parallel Processors*. ELSEVIER, 2010.

[16] A. PIAZZI, C. G. L. BIANCO, AND M. ROMANO, "ETA3-SPLINES FOR THE SMOOTH PATH GENERATION OF WHEELED MOBILE ROBOTS," *IEEE Transactions on Robotics*, VOL. 23, NO. 5, PP. 1089–1095, 2007.

[17] S. LEE AND G. KARDARAS, "COLLISION-FREE PATH PLANNING WITH NEURAL NETWORKS," IN *International Conference on Robotics and Automation*, 1997.

[18] T. BERGLUND AND A. BRODNIK, "PLANNING SMOOTH AND OBSTACLE-AVOIDING B-SPLINE PATHS FOR AUTONOMOUS MINING VEHICLES," *IEEE Transactions on Automation Science and Engineering*, VOL. 7, NO. 1, PP. 167–172, 2010.

[19] L. D. COHEN AND R. KIMMEL, "CLOBAL MINIMUM FOR ACTIVE CONTOUR MODEL: A MINIMAL PATH APPROACH," IN *Computer Vision and Pattern Recognition*, 1996.

[20] C. HAN, T. S. HATSUKAMI, AND J.-N. HWANG, "A FAST MINIMAL PATH ACTIVE CONTOUR MODEL," *IEEE Transactions on Image Processing*, VOL. 10, NO. 6, PP. 865–873, 2001.

[21] B. BANERJEE, "STRING TIGHTENING AS A SELF-ORGANIZING PHENOMENON," *IEEE Transactions on Neural Networks*, VOL. 18, NO. 5, PP. 1463–1475, 2007.

[22] J. HILGERT, K. HIRSCH, T. BERTRAM, AND M. HILLER, "EMERGENCY PATH PLANNING FOR AUTONOMOUS VEHICLES USING ELASTIC BAND THEORY," IN *Internatinoal Conference on Advanced Intelligent Mechatronics*, 2003.