Paper accepted for the 1996 IEEE International Conference on Robotics and Automation.

A review of parallel processing approaches to motion planning

Dominik Henrich

Institute for Real-Time Computer Systems and Robotics
University of Karlsruhe, D-76128 Karlsruhe, Germany
e-mail: dhenrich@ira.uka.de

Abstract

One of the many features needed to support the activities of autonomous systems is the ability of motion planning. It enables robots to move in their environment securely and to accomplish given tasks. Unfortunately, the control loop comprising sensing, planning, and acting has not yet been closed for robots in dynamic environments. One reason involves the long execution times of the motion planning component. A solution for this problem is offered by the use of highly computational parallelism. Thus, an important task is the parallelization of existing motion planning algorithms for robots so that they are suitable for highly computational parallelism. In several cases, completely new algorithms have to be designed, so that a parallelization is feasible. In this survey, we review recent approaches to motion planning using parallel computation.

1. Introduction

One of the many features needed to support the activities of autonomous systems is the ability to plan motion. Motion planning enables a robot to move in its environment securely and to accomplish a given task. In dynamic environments, the necessary adaptation of the robot motion is provided by closed control loops comprising sensing, planning, and acting, which have very short cycle times. One approach for realizing intelligent and reactive robotic systems is to integrate the planning algorithms into the control loop. Unfortunately, sound planning algorithms are complex and need long execution times. To still pursue this approach, a reduction in planning time is required. New parallel computing architectures with high computing power look promising.

One could object that the motion planning problem in general can become very complex for increasing degrees of freedom (DOF) of the robot and is still intractable, even for a parallel computer. On the other hand, the aim of parallel processing is not to reduce the intractability of complex problems, but to reduce the solution time for given problems, or to increase their solution quality. Also, it is important to approximate the general problem by a simplified, but still realistic problem. This is independent or whether sequential of parallel processing is used.

Thus, an important task is the parallelization of existing problem solutions in robotics so that they are suitable for highly computational parallelism. In several cases, fundamentally new algorithms have to be designed, so that a parallelization is feasible. The paper [28] reviews the research performed thus far in designing and implementing parallel algorithms for robotics. One of the key findings is that, in the subareas of manipulation and task planning, not much work has been published concerning parallel algorithms. Several parallel motion planing algorithms have been suggested however, which are reviewed in the rest of this paper.

2. Motion Planning Approaches

Sequential motion planning approaches have been classified in: skeletons, cell decompositions, potential fields, and mathematical programming [16]. In the *skeleton* approach, the free configuration space (free C-space), i.e., the set of feasible motions, is retracted, reduced to, or mapped onto a network of one-dimensional (1D) lines. In the *cell decomposition* approach, the free C-space is decomposed into a set of simple cells, and the adjacency relationships among the cells are computed.

To structure the parallel approaches, these keywords are modified by exchanging "skeletons" and "cell decompositions" with *graph-based* and *grid-based approaches*. The parallel ancillary algorithms, which are necessary to fulfill the planning task, are omitted due to space limitations.

2.1 Graph-based Approaches

The graph-based approaches include irregular skeletons and object-dependent cell decompositions. These approaches consist of two basic phases. The first phase is associated with the construction of a graph representing relations between free space. After computing the graph, in the second phase, the optimal path referring to a certain criterion (shortest distance, minimum time, etc.) has to be found.

Firstly, regarding the calculation of skeletons in the first phase, results obtained in parallel computational geometry can be outlined. The Voronoi diagram for a set of n points can be computed in $O(\log^2 n)$ time using n processors on a parallel random access machine (CREW-PRAM) [1]. On the same type of machine, an $O(\log n)$ time algorithm using $O(n/\log n)$ processors for computing the visible portion from a point in the plane of a simple (i.e., non-intersecting) polygonal chain with n vertices in the plane is given in [4]. Let $\mathrm{Sort}(n)$ denote the time to sort n elements. Then, the above visibility problem can be computed in $O(\mathrm{Sort}(n))$ time on an n-processor hypercube [23].

Secondly, the calculation of the object-dependent cell decomposition is regarded. In [36], the geometric structure of the configuration space obstacles is used to generate a small number of free cells, allowing the search process to work most efficiently. The algorithm has been implemented on a 16-node Transputer network, obtaining a speed-up of 11.6.

The remaining research work summarized here concentrates on the second phase of graph-based approaches. One way to do this involves the well-known graph search techniques. In [26], parallel versions of Dijkstra's method are given for shared memory CRCW and EREW machines. With n vertices in the graph, the complexity of the algorithm is $O(n \log n)$ using n processors. In the case of visibility graphs (V-graphs), the A* search has been applied on multiple cooperating mobile robots with particular priorities in [37]. The search algorithm is executed on a shared memory MIMD computer and all the robots' paths are planned simultaneously. For twelve mobile robots, collision-free paths in a 2D-workspace are computed with six processors within 18 s and a speedup of 3.74. In [43], shortest-path algorithms for cell decompositions are investigated analytically on a CREW shared memory machine. For a robot with d DOFs and a workspace with n cells, the sequential algorithm needs $O(d^2\ n^d \log n)$ steps. Using $o(d)$, $o(d^2)$, and $o(n^d)$ processors, $O(d\ n^d \log n)$, $O(n^d \log n)$, and $O(d^2\ k \log n)$ steps, respectively, are necessary to find the shortest path of length k.

Genetic algorithms can also be used to find a (sub-) optimal path in graphs. A V-graph-based approach to plan the shortest collision-free path in 3D for mobile robots is taken in [12a][1]. As the plain V-graph algorithm does not find the shortest path between 3D polyhedrons, the optimal edges are selected by a genetic algorithm. The optimal vertices along the optimal edges are computed by a "recursive compensation algorithm". A graph similar to the Voronoi diagram is used in [38]. Multiple mobile robots plan their paths independently and communicate locally to avoid competition for common free space. In the genetic algorithm, the node numbers of the skeleton are used to encode a path as a genetic string. The cross-over operation works only on paths with a common node, where the partial paths can be exchanged. The fitness function depends on the path length and the cost for commonly used free space.

Besides the graph search itself, the different processing phases occurring with graph-based approaches can be processed in parallel. For skeletons, this was examined in [35] to generate minimum-time trajectories between two points in the presence of polyhedral 3D obstacles for Cartesian robots.

2.2 Grid-based Approaches

The grid-based approaches include skeletons with a regular network of 1D lines and object-independent cell decompositions.

Similar to graph-based methods, in grid-based approaches, graph search algorithms can be used to find an optimal path. In [6][2], a hierarchical representation of the work- and C-space by bitmaps serves as the basis. In this cell decomposition, a potential field with few local minima is pre-computed using a combination of an attracting force towards the goal configuration and a Voronoi diagram. Collision free paths are planned in the hierarchical representation by a best-first search or Monte-Carlo search guided by the potential field. A parallel version of the second search method is described in [10b][3]. The hierarchical bitmap representation is broadcast to all processors, which perform a quasi-best-first search. Experiments for a seven DOF redundant robot in a 128×128×128 cell workspace were done on a Connection Machine CM-5 and needs a few seconds in average. Contrasting to the this approach, another randomized approach in [32b] avoids pre-computed heuristics and, therefore, is suited for dynamic environments. The parallel search is conducted in the discretized configuration space which needs not to be represented explicitly. The planner uses a number of rule-based sequential search processes working to find a path connecting the initial configuration to the goal via a number of randomly generated subgoal configurations to avoid deep local minima. On a cluster of 45 SUN4 and SGI machines under PVM, the average planning time for a six DOF manipulator in a cluttered environment occupied with 20 randomly sized and located rectangular obstacles is ca. 35 s. For a more realistic environment, it takes less than 10 s in average. For both above approaches, the interprocessor communication required is minimal because only problem and solution data have to be transferred once.

Besides graph search algorithms, the grid-based representations can be explored by (incomplete) genetic algorithms. In [7][4], the motion planner is based on two parallel genetic algorithms: an exploration algorithm and a search algorithm. The purpose of the exploration algorithm is to collect information about the environment with an increasingly fine resolution by placing landmarks in the searched space, thus, the C-space is not built up completely. The goal of the search algorithm is to opportunistically check if the target can be reached from any given placed landmark. Both algorithms use "Manhattan paths", which allow moves in only one DOF at a time, as genetic strings of fixed length. The fitness function minimizes the distance between the first collision and the goal configuration. Using 128 Transputers, the planning time of a six DOF robot in a environment with another not-moving robot is ca. 2 s. For an industrial application with several hundreds of geometrical entities, the evaluation of Manhattan paths is questionable. In [41], a quite uncommon grid for 2D Cspace representations is used. The genetic algorithm works over a search area enclosed by a circle, whose circumference is divided according to a predefined resolution. The angles formed by each division of the circumference represent the population. Unfortunately, the objective function which only make uses of local information may lead to bad solutions. In [21], mobile robot path planning in $2\frac{1}{2}$D grid with dynamic obstacles is investigated. In contrast to former approaches, genetic strings of variable length are used, which encode moves in one of eight possible directions. Cross-over for a path pair is based on randomly connecting points within a certain proximity. For mutation, the path remainder at a randomly selected point is replaced by a randomly generated segment. The fitness function minimizes path length,

1 This paper includes [11].
2 An earlier version of this paper is [5].
3 Earlier work is reported in [9, 10a]
4 Previous papers are [2, 3, 46, 25, 47].

traversing energy and traversing time. Finally, in [27], a triangulated terrain model including slopes and obstacles is used as a grid. A "linking method" based on the steepest descent finds partial paths between two random nodes of the skeleton. By this method, the cross-over or mutation operations join two paths at randomly selected nodes or modify existing paths, respectively. The energy consumption, including speed, slope, and friction, is used as the fitness function. With the exception of [7], the genetic algorithms have not been implemented on a parallel computer, although they imply scalable parallelism.

Another incomplete parallel motion planning approach uses neural networks. In [22], a Hopfield network for mobile robot path planning in 4- or 8-connected grids is introduced, where grid nodes and edges correspond to neurons and links, respectively. The network is initiated with an arbitrarily selected path that connects the starting location to the destination. The neurons adjust the local path in a (local) mesh to reduce the path cost. Groups of neurons, e.g., arranged like a chessboard, accomplish the global planning. Simulated annealing is used to avoid trapping at a local minimum. Although the optimal path is not guaranteed, a slow annealing rate provides a good chance of generating a nearly optimum one.

The last basic parallel method for grid-based motion planning is cellular automata. In [49], a method for solving visibility-based terrain path planning problems using massively parallel hypercube machines is proposed. A typical example is to find a path that is hidden from moving adversaries. This kind of problem can be generalized as a time-variant, constrained path planning problem and is proven to be computationally hard. An approximation based on both temporal and spatial sampling is proposed. Since a 2D grid cell representation of terrain can be embedded into a hypercube with extra links for fast communication, the method can be very efficient when implemented on hypercube machines. The time complexity is in general $O(t \cdot e \cdot \log(n))$ using $O(n)$ processors, where t is the number of temporal samples, e is the number of adversary agents, and n is the number of grid cells on the terrain. A 512×512 terrain has been implemented on the Connection Machine CM-2 with 8K processors.

2.3 Potential Field Approaches

The potential field approach uses a scalar function called the potential. It has a minimum, when the robot is at the goal configuration, and has a high value on obstacles. Everywhere else, the function slopes down toward the goal configuration, so that the robot can reach it by following the negative gradient of the potential. The high value of the potential prevents the robot from running into obstacles. Since local minima (other than the goal) are a major cause of inefficiency for potential field methods, most of the parallel versions use a global potential map as a navigation function. Additionally, such a map has the advantage of combining model-based and sensor-based workspace representations.

A very convenient method for computing a global potential map is the use of cellular automata. Using this parallel processing principle, two different approaches can be identified: wavefront expansion and relaxation. In wavefront expansion, the shortest distance to the goal in the map is computed by setting off a wave from the goal location. The wave propagates in all directions with the map cells on the wave front updating their distance to the goal and pushing the wave forward. Since the shortest path is sought, the path will always touch the border of obstacles, which have to be circumvented. In relaxation, each cell samples the potential values of its neighbors and adjusts its own potential such that it satisfies a given relaxation equation. The potential of cells representing obstacles are held constant. Therefore, the path will stay away from obstacles, although it will not be the shortest one.

Regarding the wavefront expansion by cellular automata, in [50], a sensor-based and analytical world model is assumed. In a 2D C-space, two global potential fields (one starting from the goal and one from the start position) are computed and summed up. The used SIMD computer was the ICL Distributed Array Processor (DAP), where the workspace cells are mapped onto the processors arranged in a mesh network. To compute a 64×64 potential field, 0.15 s are necessary.[5] Using only one wavefront and propagating motion directions instead of distances, in [39] the algorithm is made adaptive by utilizing a multi-resolution representation of the map. To find a path in a 32×32 map with three resolution levels, ca. 0.01 s are necessary on the AMT DAP 510. In [45], an analogy of the Huygens' principle (known in physics) is applied recursively. The meeting points of multiple wavefronts serve as the wave source (start and goal location initially) in the next recursion. In [13], visibility aspects are included in the wavefront approach in a multirobot environment. Therefore, each wavefront message includes a d-dimensional pyramid indicating the visible region from the last source. This reduces the total number of messages needed. Finally, in [44], the costs of the straight path from goal to start is used for heuristic pruning, so more expensive paths are not explored further. On the CM-2, a 1024×1024 map is processed in 5.9 s. The processing time increase as $O(l)$ with the path length l as long as there is a processor available for each map cell. Otherwise, it increases as $O(l^{2.4})$ when there are not enough processors available. Unfortunately, none of the above mentioned papers report comparisons with other parallel approaches.

Besides the wavefront algorithm itself, the problem of mapping the grid space to the processor space has been discussed.[6] The works cited so far use a simple mapping strategy that divides the grid space into partitions (of size 1×1 in all but the last reference) and map each partition to a processor. To increase the processor utilization, the grid space can be partitioned into covering rectangles, which are mapped to the processor space by using the simple mapping function [51]. When the size of the covering rectangle decreases, the processor utilization increases while the

5 To make the timings results comparable, the cases of one-to-one mapping of cells onto processors are cited.

6 This discussion refers to the maze-routing problem but the results can be applied to motion planing by global potential fields too.

number of boundary cells also increases. A generalized version keeps the size of the covering rectangle as a parameter. The optimal size of the covering rectangle has been determined by experiments. In [52], this algorithm is modified by using mirror images to allow higher processor utilization while reducing the number of boundary cells. Simulations show the improvement in an obstacle-free grid space. Additionally, a dynamic mapping algorithm is proposed which optimally maps an obstacle-free grid space.

In the following, we regard relaxation by cellular automata to calculate the global potential field map. In [30], each cell continuously samples the potential values of its neighbors and adjusts its own potential to the mean of the maximum and minimum potential found in its neighborhood. The start and goal cells have a constant high and low potential, respectively. It is stated that to reach the stable potential distribution, O(N) relaxation steps are required, where N is the number of cells in the map. The time required for the characteristic gradients to emerge is O(l), with l being the path length, though it is not clear how this length is determined. Sequential simulations on a Symbolics 3640 need 4 s for a single relaxation step in a 70×40 map. An extension to time-varying environments in [32a][7] use short- and long-term maps and avoid dynamic obstacles by computing their "bow wave". Unfortunately, this extension is not complete and the cells need global information. Another type of relaxation mechanism is used in [18][8], where the dynamic behavior of fluids in the robot workspace is simulated. The fluid flow is caused by a pump at the robots actual position and an outlet installed at its destination. The fluid flow simulation takes $\frac{1}{2}\varepsilon N$ iterations per grid cell to converge within a precision of $10^{-\varepsilon}$ and was implemented on a DAP machine. Both relaxation approaches are suitable for incompletely known and changing environments, and solve navigation through weighted regions.

Another parallel processing approach for potential field computations are neural networks. Applying this approach to robot motion planning, it is very similar to cellular automata simulating a wavefront expansion. A grid of neurons, which are interconnected only with their direct neighbors, represents the 2D workspace. On a conceptional level, the interconnection weights $w \in [0,1)$ can be used to include terrain properties [15] or different neuron (cell) distances [40]. The neuron of the goal location is set to the maximum activity and the other neurons are activated by propagating weighted output signals (wavefront). After reaching the neuron of the robot's current configuration, the neighboring neuron with maximum activation determines the next configuration. A hardware-oriented approach in [17] realizes the weights as time delays when propagating the output signal. The next robot configuration is determined by the interconnection link where the first signal comes in.

Special-purpose hardware implementations can be applied for computing global potential fields, too. Three different methods are identified: In [48, 24], a scalar electric potential field in a 2D resistive grid is used to exploit the advantages of parallel analog computation. The start configuration is modelled as a current source and the goal as an equal and opposite current sink. Obstacles are modelled as non-conducting solids in a conducting medium. This representation seems to be powerful when navigating in long, narrow corridors. Software simulations in a 2D Euclidean plane show that feasible paths for navigation are current streamlines. Nevertheless, when scaling the test chip to a full-sized grid, the discrimination between the best and the next-best path is critical. In [34], a parallel analog optical computation uses a 2D spatial light modulator on which an image of the potential field map is generated iteratively. Optically calculated fields contain no local minima, tend to produce paths centred in gaps between obstacles, and produce paths which give preference to wide gaps. The global potential field represented by N workspace cells can be computed in O($N \log N$) hybrid steps instead of O($N^{3/2}$) sequential ones. Calculation of 128×128 pixel fields is possible within a few 100 ms. Finally, in [33], a wavefront propagation technique is implemented on a linear systolic array architecture consisting of simple processing units. The algorithm computes a nearly optimal path by internal pipelining. For a 128×128 workspace about 23 ms are necessary. There are several options for computing the path from measurement of the voltages at the source node and all of its nearest neighbors (hardware gradient decent). There is some trade-off between speed of computation and the complexity of the on-chip circuitry, depending on which of these options is adopted. In summary, all three methods are mainly suitable for mobile robots because only a 2D configuration space is assumed. This excludes the application of these methods to manipulators with a greater number of DOFs.

Finally, processor farms and static task scheduling have been used as parallelization concepts to calculate global potential fields. In [42], 2D laminar fluid flow is simulated within obstacles represented by cycles. The two major computational tasks, the evaluation of a stream function at each map cell and an interpolation for specific stream values, are scheduled dynamically on a processor farm interconnected in a triple linear array. Static task scheduling is used in [14] to plan hierarchically motions for three DOF manipulators working in an unforeseen changing environment. Two wavefront processes beginning from the start and goal configuration are scheduled statically onto two processors and coordinated from a third one.

2.4 Mathematical Programming

The mathematical programming approach represents the requirement of obstacle avoidance with a set of inequalities on the configuration parameters. Motion planning is formulated, then, as a mathematical optimization problem that finds a curve between the start and goal configurations, which minimizes a certain scalar quantity. Since such an opimization is non-linear and has many inequality constraints, a numerical method is used to find the optimal solution. One approach to mathematical programming is neural nets, which can be found in [20]. Another approach for parallelizing standard optimization methods is to schedule statically selected tasks [8].

7 Earlier versions of these papers are [29] and [31], respectively.
8 Other versions of this paper are in [12b, 19].

3. Summary and Future Work

Most of the work has been done using genetic algorithms and cellular automata for grid-based and potential field approaches, respectively. Very little has been done in parallelizing mathematical programming approaches with application in robotics motion planning.

The research work in the field of parallel motion planning shows that by introducing parallel computation, the planning time can be reduced down to several seconds. With this and possible future improvements, the aim of a closed control loop with short cycle times seems to be attainable. Unfortunately, one of the following restrictions has been included in most of the parallel approaches:

- Only simple geometric object representations were used.
- The number of DOFs was reduced.
- No dynamic obstacles were regarded.

Additionally, further work should aim at a comparison of the different parallel approaches, especially the different wavefront expansion methods by cellular automata. Furthermore, implementations of grid-based genetic algorithms have not been done on parallel computers. The adaptation is not trivial, because it is not clear how genetic evolution changes when different interprocessor communication patterns are used. Finally, future approaches may focus on explicitly including motion time and speed in the optimization.

Acknowledgements

The work was performed at the Institute for Real-Time Computer Systems and Robotics, Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann, University of Karlsruhe. Many thanks to my colleagues Dr. Xiaoqing Cheng, Petra Bohner, and Michelle Specht for proofreading.

References

[1] Aggarwahl A., et al., "Parallel computational geometry". In: Algorithmica, vol 3, pp 293-327, 1988.

[2] Ahuactzin, J.M.; Talbi, E.-G.; Bessiere, P.; Mazer, E., "Using genetic algorithms for robot motion planning". In: Geometric Reasoning for Perception and Action. Workshop, 1991, 16-17 Sept., pp 84-93.

[3] Ahuactzin J.M., Talbi E.-G., Bessiere P., Mazer E., "Using genetic algorithms for robot motion planning". In: ECAI 92, 10th European Conference on Artificial Intelligence Proceedings, 1992, 3-7 Aug., pp 671-5.

[4] Atallah M. J., Zehn D. Z., "An optimal parallel algorithm for the visibility of a simple polygon from a point". In: Annual Symposium on Computational Geometry, pp 114-122, ACM, 1989.

[5] Barraquand J., Latombe J.-C., "Robot motion planning: A distributed representation approach". In: Technical Report, STAN-CS-89-1257, Leland Stanford Junior University, Department of Computer Science, May 1989.

[6] J. Barraquand, J.-C. Latombe, "Robot motion planning: A distributed representation approach". In: The Int. Jour. of Robotics Research, vol 10, no 6, pp 628-649, Dez. 1991.

[7] Bessière P., et al., "The Adriane's clew algorithm", In: Algorthmic Foundation of Robotics, Ed. by K. Goldberg, et al., A.K. Peters, 1994.

[8] Cela A.; Hamam Y.; Georges D., "Decomposition method for the constrained path planning of articulated systems". In: 91 ICAR. Fifth International Conference on Advanced Robotics. Robots in Unstructured Environments, Pisa, Italy, 1991, pp 994-9.

[9] Challou D.J.; Gini M.; Kumar V., "Towards real-time motion planning". In: H. Kitano et al. (eds), Parallel Processing for Artificial Intelligence, 2. Elsevier, 1994.

[10a] Challou D.J.; Gini M.; Kumar V., "Parallel search algorithms for robot motion planning". In: IEEE Int. Conf. on Robotics and Automation, Atlanta, GA, USA, 1993, 2-6 May, pp 46-51, vol 2.

[10b] Challou D.J.; et al., "A parallel formulation of informed randomized search for robot motion planning problems". In: IEEE Int. Conf. on Robotics and Automation, Nagoya, Japan, 1995, May 21-27, pp 709-714.

[11] Chung C.H.; Lee K.S., "Hopfield network application to optimal edge selection". In: 1991 IEEE International Joint Conference on Neural Networks, Singapore, 1991, 18-21 Nov., pp 1542-7.

[12a] Chung C.H.; Lee K.S., "Neural network application to the obstacle avoidance path planning for CIM computer integrated manufacturing". In: Proceedings IROS '91. IEEE/RSJ Int. Workshop on Intelligent Robots and Systems '91. Intelligence for Mechanical Systems, Osaka, Japan, 1991, 3-5 Nov., pp 824-8.

[12b] Decuyper J., Keymeulen D., "A reactive robot navigation system based on a fluid dynamics metaphor". In: Parallel Problem Solving from Nature. 1st Workshop, PPSN, 1 Proceedings, 1990, 1-3 Oct., Eds: Schwefel, H.-P.; Manner, R., pp 356-62.

[13] Dubash R.M.; Bastani F.B., "A hybrid architecture for mobile robots based on decentralized, parallel path planning". In: Proceedings ISADS 93. International Symposium on Autonomous Decentralized Systems, Kawasaki, Japan, 1993, 30 March-1 April, pp 206-14.

[14] Fink B.; Wend H.-D., "Collision-free motion-planning for robot-manipulators working in a changing environment". In: Automatisierungstechnik, vol 39, no 6, 1991, June, pp 197-200.

[15] Huse S. M., "Path analysis using a predator-prey neural network paradigm". In: Proc. of the 3rd Int. Conf. on Industrial and Engineering, 1990, pp 1054-1062.

[16] Hwang Y. K., Ahuja N., "Gross motion planning – A survey". In: ACM Computing Surveys, vol 24, no 3, Sept 1992.

[17] Kalyayev I.A., "Homogeneous neuronlike structures for optimization variational problem solving". In: PARLE '93 Parallel Architectures and Languages Europe. 5th Int. PARLE Conf. Proc., 1993, 14-17 June, Eds.: Bode A.; Reeve M.; Wolf G., Springer-Verlag Berlin, Germany, pp 438-51.

[18] Keymeulen D.; Decuyper J., "A flexible path generator for a mobile robot". In: 91 ICAR. Fifth Int. Conf. on Advanced Robotics. Robots in Unstructured Environments, Pisa, Italy, 1991, pp 1069-73.

[19] Keymeulen D.; Decuyper J., "The fluid dynamics applied to mobile robot motion: The stream field method". In: IEEE Int. Conf. on Robotics and Automation, 1994, pp 378-385.

[20] Lemmon M., "2-Degree-of-freedom Robot Path Planning using Cooperative Neural Fields". In: Neural Computation, vol 3, no 3 pp 350-362.

[21] Leung C.H.; Zalzala A.M.S., "A genetic solution for the motion of wheeled robotic systems in dynamic environ-

ments". In: Int. Conf. on Control '94, Coventry, UK, 1994, 21-24 March, pp 760-4.
[22] Lin C.-S.; Wann C.-D., "A parallel processing model for robot path planning on grid terrains". In: Int. Jour. of Robotics & Automation, vol 6, no 1, 1991, pp 1-11.
[23] MacKenzie P. D., Stout Q. F., "Asymptotically efficient hypercube algorithms for computational geometry". In: Third Symposium on the Fontiers of Massively Parallel Computation, pp 8-11, College Park, MD., Oct. 1990.
[24] Marshall G.F.; Tarassenko L., "Robot path planning using VLSI resistive grids". In: Third Int. Conf. on Artificial Neural Networks, Brighton, UK, 1993, 25-27 May, pp 163-7.
[25] Mazer E.; Ahuactzin J.M.; Talbi E.; Bessiere P., "Robot motion planning with the Ariadne's clew algorithm". In: Intelligent Autonomous Systems. IAS-3. Proc. of the Int. Conf., Pittsburgh, PA, USA, 1993, 15-18 Feb., Eds.: Groen F.C.A.; Hirose S.; Thorpe C.E., pp 196-205.
[26] Paige R. C., Kruskal C. P., "Parallel algorithms for shortest path problems". In: IEEE, 1985, pp 14-20.
[27] Pinchard O., Liegeois A., Emmanuel T., "A genetic algorithm for outdoor robot path planning". In: Intelligent Autonomous Systems (IAS-4), Eds: U. Rembold, Karlsruhe, Germany, March 27-30, 1995, pp 413-419.
[28] Prasanna V.K.; Rao A.S., "Parallel algorithms for robotics – A survey". In: Computer Science and Informatics, vol 22, no 1, 1992, July, pp 1-18.
[29] Prassler E.; "Electrical networks and a connectionist approach to path-finding". In: Proc. of the Int. Conf. "Connectionism in Perspective", Amsterdam, Elsevier, 1989.
[30] Prassler E.; Milios E., "Parallel path planning in unknown terrains". In: Proc. of the SPIE - The Int. Society for Optical Engineering, vol 1388, 1990, pp 2-13.
[31] Prassler E.; Milios E., "Motion planning amongst arbitrarily moving unkown objects". In: Proc. of the IEEE/RSJ/GI Int. Conf. on Intelligent Robots and Systems IROS'94, Munich, Germany, 1994.
[32a] Prassler E., "Robot navigation: A simple guidance system for a complex changing world". In: Environment Modeling and Motion Planning forAutonomous Robots, Eds: H. Bunke et al., World Scientific, 1995 – *to appear*.
[32b] Qin C., Henrich D., "Randomized parallel motion planning for robot manipulators", Technical Report, Computer Science Department, University Karlsruhe, 1996.
[33] Ranganathan N.; Parthasarathy B.; Hughes K., "A parallel algorithm and architecture for robot path planning". In: Proc. Eighth Int. Parallel Processing Symposium, Cancun, Mexico, 1994, 26-29 April, Eds.: Siegal, H.J., pp 275-9.
[34] Reid M.B., "Path planning using optically computed potential fields". In: Proc. IEEE Int. Conf. on Robotics and Automation, Atlanta, GA, USA, 1993, 2-6 May, pp 295-300.
[35] Rovetta A.; Sala R., "Robot motion planning with parallel systems". In: Proc.. 1992 IEEE Int. Conf. on Robotics And Automation, Nice, France, 1992, 12-14 May, pp 2224-9.
[36] Schmidt-Brauns R.; Swietlik A.; Dillmann R., "Robot path planning on transputer networks". In: Applications of Transputers 3. Proc. of the Third Int. Conf. on Applications of Transputers, Glasgow, UK, 1991, 28-30 Aug., Eds.: Durrani, T.S.; Sandham, W.A.; Soraghan, J.J.; Forbes, S.M.,pp 174-9.
[37] Shang W.; Egan G.K., "Mobile robot path planning using parallel computer system". In: IEEE Int. Workshop on Emerging Technologies and Factory Automation - Technology for the Intelligent Factory - Proc., Melbourne, Vic., Australia, 1992, 11-14 Aug., Eds.: Zurawski, R.; Dillon, T.S., pp 676-80.
[38] Shibata T.; Fukuda T., "Coordinative behavior by genetic algorithm and fuzzy in evolutionary multi-agent system". In: Proc. IEEE Int. Conf. on Robotics and Automation, Atlanta, GA, USA, 1993, 2-6 May, pp 760-5.
[39] Shu C.; Buxton H., "A parallel path planning algorithm for mobile robots". In: BMVC90 Proc. of the British Machine Vision Conf., Oxford, UK, 1990, 24-27 Sept., pp 383-8.
[40] Siemiatkowska B., "A Highly Parallel Method for Mapping and Navigation of An Autonomous Mobile Robot". In: Proc. IEEE Int. Eonference on Robotics and Automation, 1994, pp 2796-2801.
[41] Solano J.; Jones D.I., "Generation of collision-free paths, a genetic approach". In: IEE Colloquium on 'Genetic Algorithms for Control Systems Engineering', London, UK, 1993, 28 May, pp 5/1.
[42] Solano J.; Jones D.I., "Parameter determination for a genetic algorithm applied to robot control". In: Int. Conf. on Control '94, Coventry, UK, 1994, 21-24 March, pp 765-70.
[43] Stifter S., "Shortest non-synchronized motions parallel versions for shared memory CREW models". In: Parallel Computation. Second Int. ACPC Conf. Proc., Gmunden, Austria, 1993, 4-6 Oct., Eds.: Volkert, J., pp 87-104.
[44] Stiles P. N., Glickstein I. S., "Higliy parallelizable route planner based on cellular automata algorithms". In: IBM Jour. Research and Developement, vol 38, no 2, pp167-181, March 1994.
[45] Suzuki H.; Arimoto S., "Parallel-processable recursive and heuristic method for path planning". In: Proc. IROS '91. IEEE/RSJ Int. Workshop on Intelligent Robots and Systems '91. Intelligence for Mechanical Systems, Osaka, Japan, 1991, 3-5 Nov., pp 616-18.
[46] Talbi E.G.; Bessiere P., "Parallel robot motion planning in a dynamic environment". In: Second Join Int Conference on Vector and Parallel Processing", 1992, Sept, Lyon, France.
[47] Talbi E.-G.; Muntean T., "Designing embedded parallel systems with parallel genetic algorithms". In: IEE Colloquium on 'Genetic Algorithms for Control Systems Engineering', London, UK, 1993, 28 May, pp 7/1.
[48] Tarassenko L., Blake A., "Analogue computation fo collison-free paths". In: IEEECROB, 1991, pp 540-545.
[49] Teng Y.A.; DeMenthon D.; Davis L.S., "Stealth terrain navigation". In: IEEE Trans. on Systems, Man and Cybernetics, vol 23, no 1, 1993, Jan.-Feb., pp 96-110.
[50] Witkowski C. M., "A parallel processor algorithm for robot route planning". In: Int. Joint Conf. on Artificial Intelligence (IJCAI), Karlsruhe, West Germany, Aug. 1983, pp 827-829.
[51] Won Y., Sahni S., "Maze routing on a hypercube multiprocessor computer". In: Proc. of the 1987 Int. Conf. on Parallel Processing, pp 630-637,1987.
[52] Yen I.-L.; Dubash R.M.; Bastani F.B., "Strategies for mapping Lee's maze routing algorithm onto parallel architectures". In: Proc. of Seventh Int. Parallel Processing Symposium, Newport, CA, USA, 1993, 13-16 April, pp 672-9.