IN: JOURNAL OF INTELLIGENT AND ROBOTIC SYSTEMS,
VOL 20, NO 1, PP 45-69, SEPT 1997.

Fast motion planning by parallel processing – A review

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. As a classification scheme, we use the structure given by the different approaches to the robot's motion planning. For each approach, the available parallel processing methods are discussed. Each approach is uniquely assigned a class. Finally, for each referenced research work, a list of keywords is given.

Keywords: robotics; autonomous systems; parallel algorithms; parallelism and concurrency; motion planning; review;

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 (within milliseconds). 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 computing architectures with high computing power look promising.

In spite of significant improvements in the processing speed, sequential processors are far from rendering sufficient computing capacity for an advanced robotic planning system. On the

other hand, modern VLSI technology offers a unique opportunity to close this gap by parallel computing. One could object that massively parallel computers do not serve as a conceivable platform for motion planning due to their high cost and limited availability. However, it can be expected that the progress in the design of new VLSI circuits and the reduction in component cost will make the highly parallel machines available, very probably in the next decade, it will be possible to build highly parallel computers with relatively small costs.

Also, it can be said that the motion planning probblem 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 of whether sequential or parallel processing is used.

Today's sequential computers may be sped up only through intensive technological effort because the performance is physically limited by these architectures. Computational parallelism is one solution to this problem. By adding processing units in parallel computers, the process time can be arbitrarily sped up for *corresponding* complex problems.[1] On the other hand, the available computational parallelism has to be exploited in an efficient way. The solution methods of different applications can be parallelized in various ways. An improvement in performance cannot be achieved by solely increasing the number of processing units because the time for necessary communication or for additional data administration may increase simultaneously.

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 [Prasanna92] 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 this paper.

The rest of the paper is organized as follows: First, we introduce the classification scheme for the parallel motion planning approaches in Section 2. Then, the results of the single motion planning approaches are presented and discussed in Section 3. After summarizing the work done so far, open problems and interesting future research fields are pointed out in Section 4. The paper ends with a list of references with the corresponding keywords.

1 The solution time for a *fixed-sized* problem can asymptotically be reduced by parallel processing at most down to the inherent sequential fraction of the solution algorithm (Amdahls law) [Gustafson88].

2. Classification Scheme

For classifying individual research areas, a set of keywords is given in Table 1. The keywords are chosen from the two combined research fields, robotics and parallel processing. Additionally, the keywords are distinguished by the aspect they concern in the field, i.e. robotics application or robotics motion planning approaches. Most of the keywords are self-explanatory; the others will be explained later.

Field:	Aspect:	Keywords:
Robotics	Applications	Manipulators; Telerobotics; Mobile robots; Grippers; Microrobotics; Multirobot environments
	Motion planning approaches	Skeleton; Cell decomposition; Potential field; Mathematical programming
	Ancillary algorithms	C-space computation; Collision avoidance; Path optimization
Parallelism	Approaches	Graph search; Genetic algorithms; Cellular automata; Neural nets; Processor farms; Static task scheduling
	Architectures	MIMD computers; SIMD computers; Dataflow machines; Shared memory systems; Special purpose hardware; No parallel implementation
	Interconnections	Bus networks; Linear networks; Ring networks; Mesh networks; Torus networks; Hypercube networks; Special purpose networks
	Scalability	Scalable parallelism; Specialist parallelism

Table 1: Selected keywords for the two combined research fields, robotics and parallelism

2.1 Robotics

The keywords of the aspect "Motion planning approaches" and their definitions are derived from the review paper in [Hwang92]. The keywords represent a list of basic approaches for gross motion planning. These approaches are not necessarily mutually exclusive, and a combination of them is often used in developing a motion planner.

In the *skeleton* approach, the free configuration space (C-free-space), i.e., the set of feasible motions, is retracted, reduced to, or mapped onto a network of one-dimensional (1D) lines. This approach is also called the retraction, road map, or highway approach. The search for a solution is limited to the network, and motion planning becomes a graph-searching problem. The well-known skeletons are the visibility graph (V-graph) and the Voronoi diagram, commonly used ones for the 2D, the silhouette and the subgoal network.

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. A collision-free path between the start and the goal configuration of the robot is found by first identifying the two cells containing the start and the goal and then connecting them with a sequence of connected cells.

The *potential field* approach constructs a scalar function called the potential that has a minimum when the robot is at the goal configuration, and a high value on obstacles. Everywhere else, the function is sloping down toward the goal configuration, so that the robot can reach the goal

configuration by following the negative gradient of the potential. The high value of the potential prevents the robot from running into obstacles.

The *mathematical programming* approach represents the requirement of obstacle avoidance with a set of inequalities on the configuration parameters. Motion planning is then formulated as a mathematical optimization problem that finds a curve between the start and goal configurations minimizing a certain scalar quantity. Since such an optimization is non-linear and has many inequality constraints, a numerical method is used to find the optimal solution.

2.2 Parallelism

The keywords in the parallelism aspects "approaches", "architectures", and "interconnection" are well-known and need not be described any further.[2] Additionally, we apply the aspect "scalability" containing two keywords. The keyword *specialist parallelism* refers to approaches where the problem is partitioned into multiple distinguished tasks and each of them is solved by a separate process. The keyword *scalable parallelism* refers to approaches where the problem is partitioned into similar tasks and, additionally, the number of tasks can be chosen freely within a wide range. In the second kind of parallelism, by increasing the number of processing units, either the solution time can be sped up or the solution quality can be increased. In both cases, the efficient exploitation of the computational power supplied by the processors has to be regarded.

3. Motion Planning Approaches

To structure the parallel approaches, the classification of motion planning approaches as given in Section 2.1 is used. For each basic approach, the available parallel processing methods are discussed. Each parallel method is uniquely assigned to a class. Segmentation of the field according to the type of problem, where several problems may be solved by the same (sequential or parallel) approach, was rejected

In contrast to the classification, the basic data representation of skeletons and of cell decomposition can be subdivided into (multi-dimensional) grids and graphs. Additionally, the parallel algorithms depend much more on the data representations than on the way these representations have been retrieved. Therefore, the classification scheme given by the four keywords for general motion planning approaches is modified by exchanging the classes "skeletons" and "cell decomposition" with *graph-based* and *grid-based* approaches. The graph-based approaches include irregular skeletons and object-dependent cell decompositions. The grid-based approaches include skeletons with a regular network of 1D lines and object-independent cell decompositions.

---

[2] In the field parallelism, we use only the most common kewords resp. methods. Further keywords exist especially for the aspect "interconnections" indicating more complex topologies, which are not needed here.

The following subsections summarize parallel motion planning approaches according to the modified structure of general motion planning: graph-based approaches, grid-based approaches, potential fields, mathematical programming, and ancillary algorithms.

3.1 Graph-based Approaches

The graph-based approaches include irregular skeletons and object-dependent cell decompositions. For examples see Figure 1. In this case, all collision-free paths are stored explicitly and cannot be generated implicitly. 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.

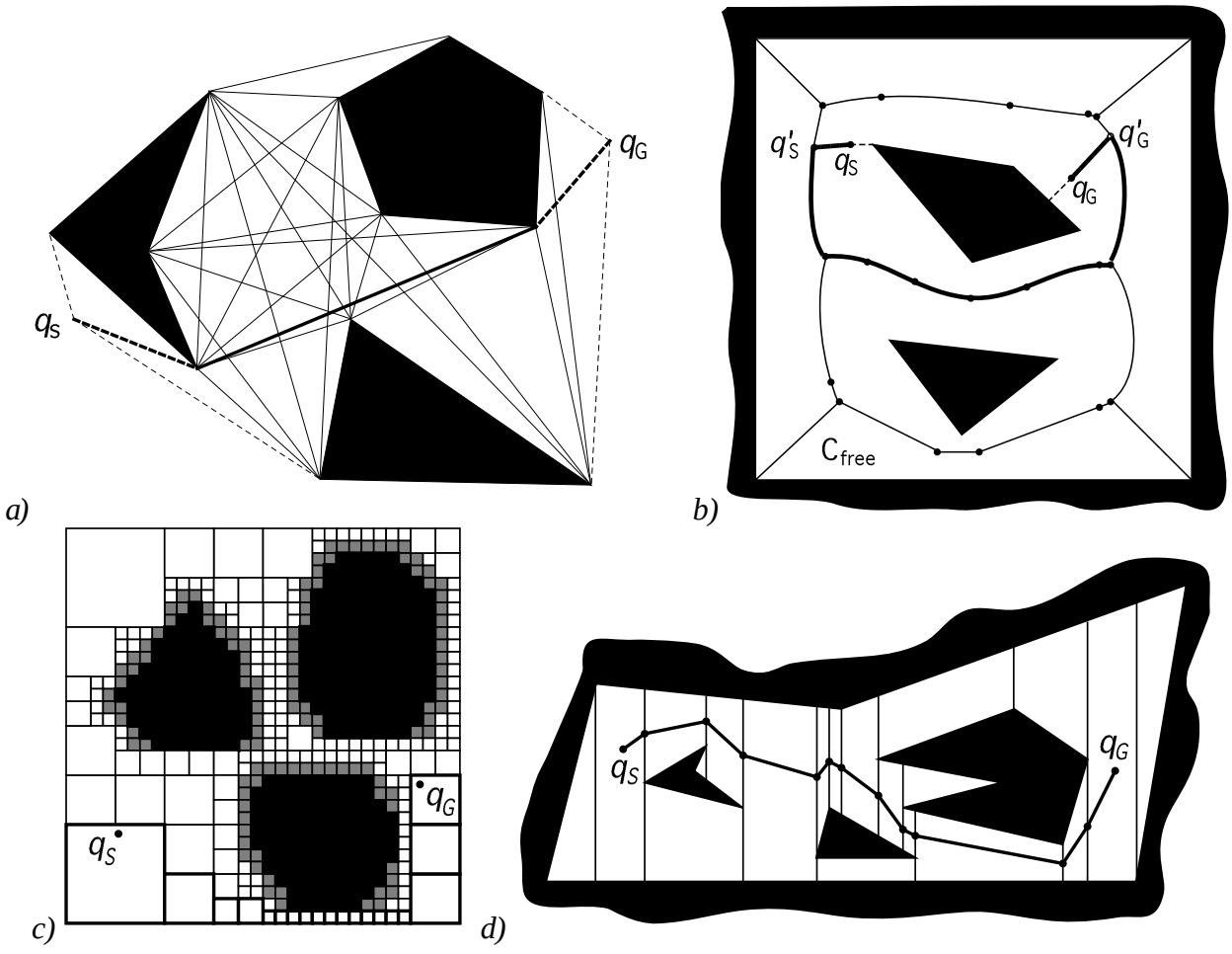


Figure 1: Different types of graph based environment representations [Latombe91]: a)V-graph, b) Voronoi diagram, c) quadtree, d) trapezoid cell decomposition

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 CREW[3] Parallel Random Access Machine

3 CREW: Concurrend Read Exclusive Write

(PRAM) [Aggarwal88]. 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 [Atallah89]. Let Sort(n) denote the time to sort n elements. Then, the above visibility problem can be computed in O(Sort(n)) time on an n-processor hypercube [MacKenzie90].

Secondly, the calculation of the object-dependent cell decomposition is regarded. In [SchmidtBrauns91], 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 [Paige85], 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 search method A* has been applied on multiple cooperating mobile robots with particular priorities in [Shang92]. The search algorithm is executed on a shared memory MIMD[4] 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 [Stifter93], 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.

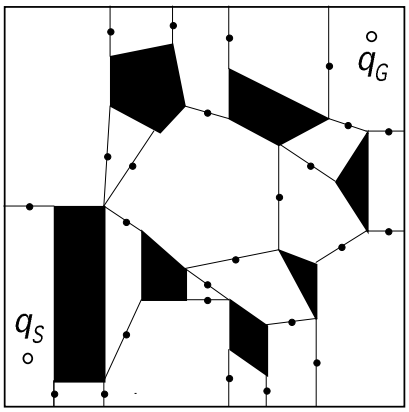
Genetic algorithms can also be used to find a (sub-) optimal path in graphs. A V-graph-based approach to plan the shortest collision-free path in 3D for mobile robots is taken in [Chung91b][5]. As the plain V-graph algorithm does not find the shortest path between 3D polyhedrons, the optimal edges are selected by a genetic algorithm. The optimal vertices along the optimal edges are computed by a "recursive compensation algorithm" described in an earlier paper.

An example for genetic algorithms used to find paths in graphs is given in Figure 2. Here, a 2D graph similar to the Voronoi diagram is used [Shibata93]. 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 paths of the child string can be improved by deleting posible cycles. The mutation operation is a random continuation starting from a random node in the path. The fitness function depends on the path length.

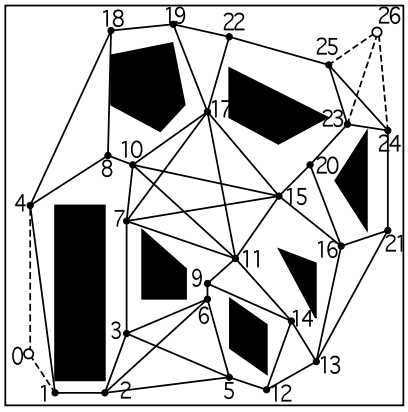
4 MIMD: Multiple Instructions Multiple Data

5 This paper includes [Chung91a].

a)



b)



c)

Path encoded as gen string:
0-4-8-10-11-17-22-25-26, with 0 = start node, 26 = goal node

Cross-over operation:
*parent1 = 0-4-8-**10**-11-15-17-22-25-26*
*parent2 = 0-1-2-6-9-11-7-**10**-8-18-19-22-25-26,*
*child1 := 0-4-8-**10**-8-18-19-22-25-26*
*child2 := 0-1-2-6-9-11-7-**10**-11-15-17-22-25-26*

Delete cycles:
*child2 = 0-1-2-6-9-**11**-7-10-**11**-15-17-22-25-26*
*child2' := 0-1-2-6-9-**11**-15-17-22-25-26*

Mutation operation:
*child2' = 0-1-2-6-9-11-**15**-17-22-25-26*
*child2'' = 0-1-2-6-9-11-**15**-20-23-26*

Fitness function F:
F*(child2'')* = $d_{0,1} + d_{1,2} + d_{2,6} + d_{6,9} + \ldots + d_{23,26}$, *with distance* $d_{i,j}$ *between nodes* i *and* j

Figure 2: Example for parallel motion planning by genetic algorithms in a graph [Shibata93]. a) Construction of the graph nodes b) The nodes in each free area are connected to represent the graph links c) Typical genetic operations based on the graph in b)

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 [Rovetta92] to generate minimum-time trajectories between two points in the presence of polyhedral 3D obstacles for Cartesian robots. The algorithm can be compared to the V-graph method, where the objects, which intervene between the robot and its goal, are incorporated into a single mono-obstacle. The whole process is divided into three successive stages: growing obstacle set (construction of the mono-obstacle), search for the trajectory and choice of the optimal trajectory. For the first two stages, specific tasks are identified and scheduled on up to four processing units, so that the number of processors cannot be scaled up.

3.2 Grid-based Approaches

The grid-based approaches include skeletons with a regular network of 1D lines and object-independent cell decompositions. For examples see Figure 3. In contrast to the graph-based

approaches, here, the free C-space is not precomputed explicitly but can be searched implicitly using a collision test.

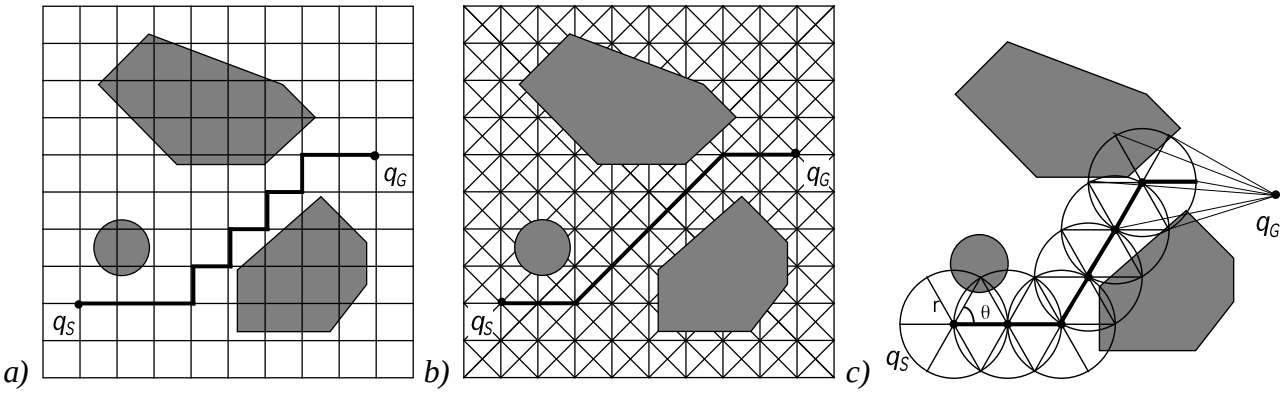


Figure 3: Different types of grids for parallel motion planning: a) 4-conneted grid, b) 8-connected grid and c) "circular grid". In the "circular grid", the neighbouring nodes lie on a circumference with radius r *divided according to a predefined resolution* Θ *[Solano93].*

Similar to graph-based methods, in grid-based approaches, graph search algorithms can be used to find an optimal path. In [Barraquand91][6], 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 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 Monte-Carlo search is described in [Challou95][7]. 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 [Qin96b][8] avoids pre-computed heuristics and, therefore, is suited for dynamic environments (see Figure 4). 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[9], the average planning time for a six DOF polyhedral 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.

6 An earlier version of this paper is [Barraquand89].

7 Earlier work is reported in [Challou93a, Challou93b]

8 A more detailed description is given in the technical report [Qin96a].

9 Parallel Virtual Machine (PVM) is a parallel programming interface based on the message-passing paradigm

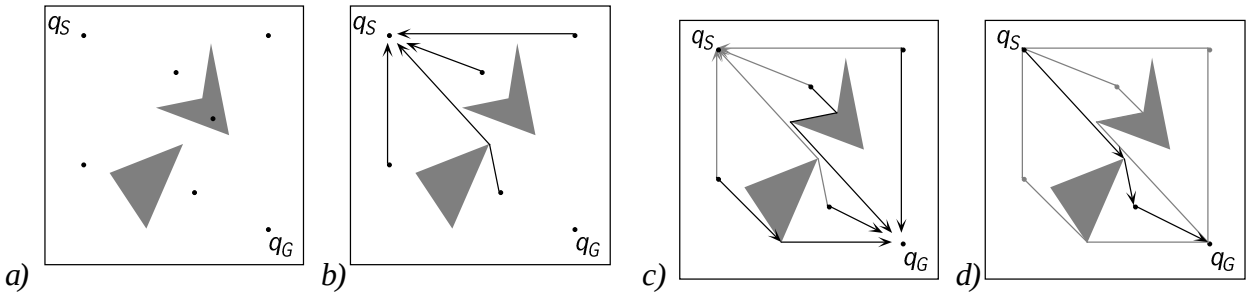


Figure 4: Example for randomized parallel motion planning in a grid [Qin96]: a) choose random subgoals in parallel, b+c) search in grid first and second subpaths in parallel, and d) select best path

Besides graph search algorithms, the grid-based representations can be explored by (incomplete) genetic algorithms (see Figure 5). In [Bessiere94][10], 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 goal 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 represented by boxes 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 [Solano93], 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 (see Figure 3). 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 [Leung94], 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, traversing energy and traversing time.

Finally, in [Pinchard95], a triangulated terrain model including slopes and obstacles is used as a grid. A "linking method" uses the steepest-descent algorithm to find a partial path 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

[10] Previous papers are [Ahuactzin91, Ahuactzin92, Talbi92, Mazer93, Talbi93, Bessiere94].

[Bessiere94], the genetic algorithms have not been implemented on a parallel computer, although they imply scalable parallelism.

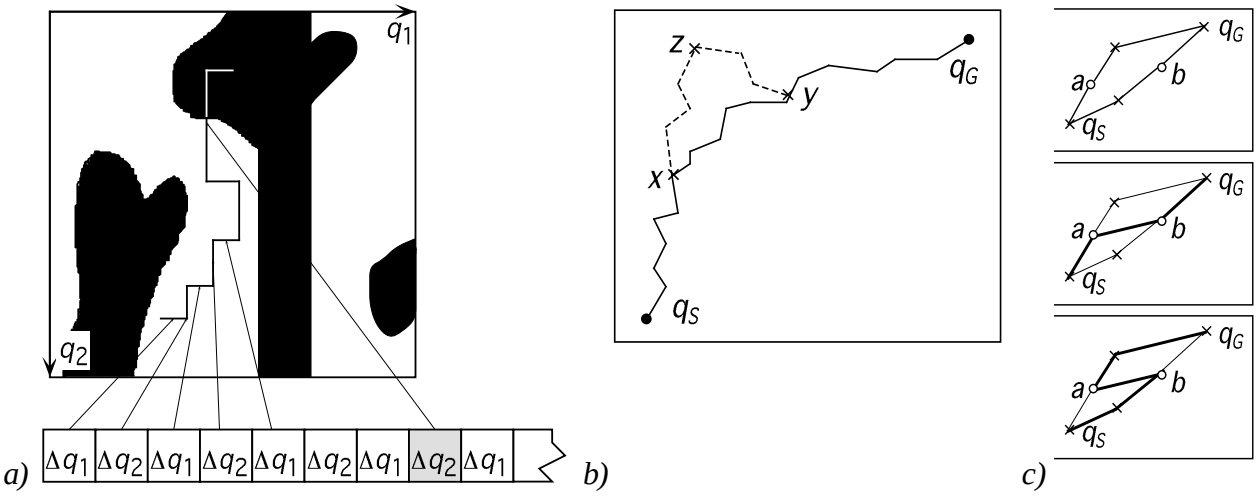


Figure 5: Examples for the basic operations of genetic algorithms for parallel motion planning in a grid [Bessiere94, Pinchard95]. a) Strings encode the length of moves in one of the available grid directions at a time. b) The mutation operation modifies the subpath between x *and* y *by a path via* z *(*x, y, z *are choosen randomly) c) The cross-over operation interchanges the first and second halves of two paths at two randomly choosen points* a *and* b *generating two new paths.*

Another incomplete parallel motion planning approach uses neural networks (see Figure 6). In [Lin91], 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 start to the goal configuration. 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 [Teng93], 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.

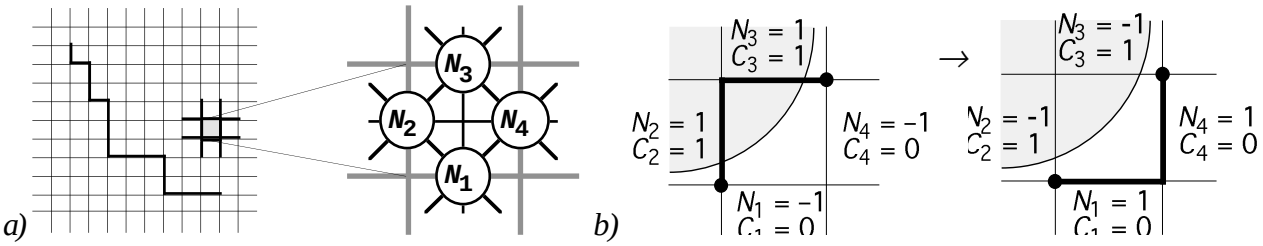


Figure 6: a) Example for parallel motion planning in a grid by neural networks [Lin91]. b) In the local Hopfield networks the obstacles are represented by the cost C_i*=1 of the links,* C_i*=0 otherwise. The states of the neurons* N_i *indicate whether a link is part of path (*N_i*=1) or not (*N_i*=–1). Using as costs function –(*$C_1N_1 + C_2N_2 + C_3N_3 + C_4N_4$*), the state transition is determined by* $N_i(t+1) = Sign(costs) * N_i(t)$ *for all neurons in a local mesh (*$i=1,\ldots,4$*)*

3.3 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 (see Figure 7). 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.

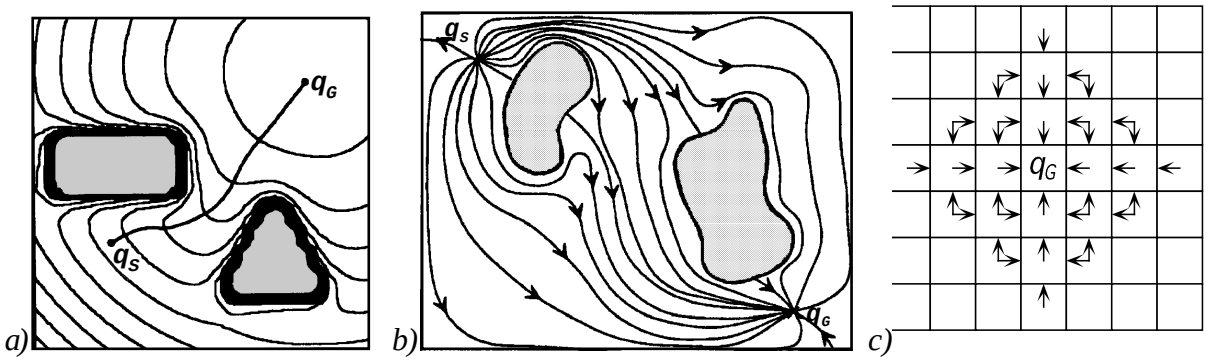


Figure 7: Different types of global potential field maps encoding a) minimum distance to the goal, b) simulated current flow, and c) propagated move directions [Latombe91, Keymeulen91, Shu90]

Regarding the wavefront expansion by cellular automata, in [Witkowski83], 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 configuration) are computed and summed up. The used SIMD[11] 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.[12] Using only one wavefront and propagating motion directions instead of distances, in [Shu90] 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 [Suzuki91], 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 configuration initially) in the next recursion. In [Dubash93], 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 [Stiles94], 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.

All of the above cellular automata wavefront expansions need O(l) steps to find one of the shortest paths of length l. Using a d-dimensional potential field map with n cells per dimension, the maximum path length is bounded by O(n^d). This is also the number of steps needed to compute the complete potential field map disregarding the shortest path length. In such a complete map, multiple shortest paths starting from different configurations can be determined without recomputation of the potential field.

11 SIMD: Single Instruction Multiple Data

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

Besides the wavefront algorithm itself, the problem of mapping the grid space to the processor space has been discussed.[13] 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 (see Figure 8). 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 [Won87]. When the size of the covering rectangle decreases, the processor utilization increases while the 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 [Yen93], 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.

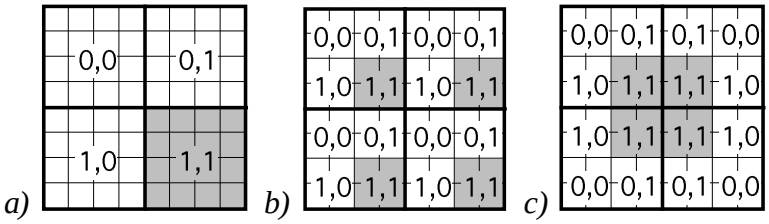


Figure 8: Different mappings of a potential field map onto four processors in the 2D grid (i,j), i,j ∈ {0,1}*: a) blockwise, b) blockwise cyclic, and c) mirrored blockwise cyclic [Yen93]*

In the following, we regard relaxation by cellular automata to calculate the global potential field map (see Figure 9). In [Prassler90], 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 [Prassler95][14] 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 [Keymeulen91][15], where the dynamic behavior of fluids in the robot C-space is simulated. The fluid flow is caused by a pump at the robots start configuration and an outlet installed at its goal configuration. The fluid flow simulation takes $\frac{1}{2}\varepsilon n$ iterations per grid cell to converge within a precision of $10^{-\varepsilon}$ and was

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

14 Earlier versions of these papers are [Prassler89] and [Prassler94], respectively.

15 Other versions of this paper are [Decuyper90, Keymeulen94].

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 [Huse90] or different neuron (cell) distances [Siemiatkowska94]. 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 [Kalyayev93] 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.

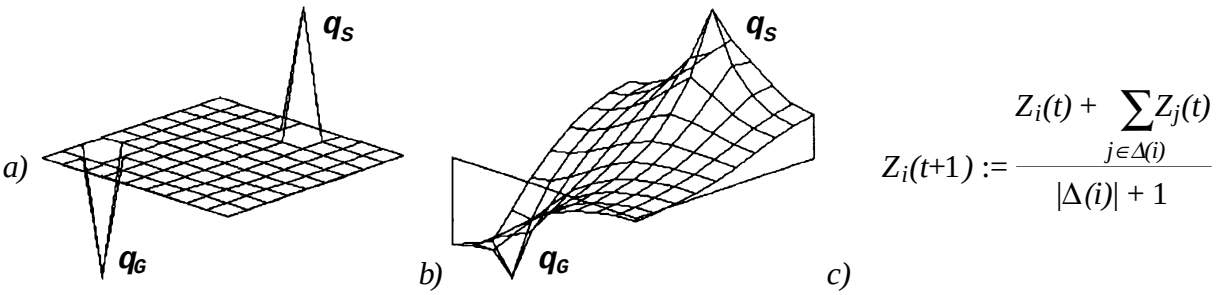


Figure 9: Example for generating a global potential map by relaxation using cellular automata [Prassler89]. a) Initial map with start and goal configurations q_S *and* q_G *b) Final map c) Update formula: Each cell* i *adapts its potential* Z_i *according to the potential of its set of direct neighbours* $\Delta(i)$*.*

Special-purpose hardware implementations can be applied for computing global potential fields, too. Three different methods are identified: In [Tarassenko91, Marshall93], 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 [Reid93], 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.

In [Ranganathan94], 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.

Finally, processor farms and static task scheduling have been used as parallelization concepts to calculate global potential fields. In [Solano94], 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. By reordering the tasks according to execution time, a 41×200 map could be computed on eight T800 Transputers with a speedup of a factor of 5.8 in 89 s. Static task scheduling is used in [Fink91] 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. Another approach distributes the path sections of the path planned on a coarse level onto different processors, which improve the path on a fine representation level.

In summary, most of the potential field methods are 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.

3.4 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 optimization 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. In [Lemmon91], a neural network solution to path planning by two DOF robots is proposed. The network is a 2D field of neurons, which forms a distributed representation of robot's workspace. Lateral interconnections between neurons are cooperative, so that the field exhibits oscillatory behavior. It is shown, how the oscillatory behavior can be used to find the dynamic programming solution of the path planning problem.

Another approach for parallelizing standard optimization methods is to schedule statically selected tasks. In [Cela91], an optimal path planning approach is proposed for industrial robots based on non-linear programming, decomposition-coordination and feedback decoupling and linearization. The path is obtained through the optimization of a time-energy criterion for the linearized model by static feedback. Decomposition of the original optimization problem at the joint

levels makes parallel processing possible and reduces the CPU time needed. The method does not suffer from curse of dimensionality. Nevertheless, robot segments are approximated by cylinders with spherical caps.

3.5 Ancillary Algorithms

Besides the different parallel motion planning approaches themselves, various ancillary algorithms are necessary to fulfill the planning task. In this section, we present parallel versions of some of these additional algorithms.

The fundamental prerequisite of all motion planning algorithms using an explicit C-space representation is the transformation of the obstacle into C-space. In [Dehne89], a systolic algorithm for computing the C-space of an arrangement of arbitrary obstacles in the plane for a rectilinearly convex robot is presented. The obstacles and the robot are assumed to be represented in digitized form by a $n \times n$ binary image. The algorithm is designed for a mesh network with $n \times n$ processors (using the canonical representation of an image on a processor array) and has an execution time of $O(n)$, which is asymptotically optimal. A similar problem for arbitrarily shaped robots with intersection joint axes is addressed in [LozanoPerez91]. Thereby, it is recognized that one can compute a family of primitive maps, which can be combined by superposition based on the distribution of real obstacles. The algorithm runs on a Connection Machine with 8192 processors. In [Gonzalez91], the C-space obstacles for a n DOF robot are approximated by sets of n–1-dimensional slices, recursively built up from 1D slices. The tasks to compute these slice projections have a tree-like dependency and are scheduled onto a (triple) linear array, star and tree topology. For a two-link manipulator, a speedup of about a factor of 11 can be obtained using 12 Transputers. Finally, in [Kavraki93], the approach to this problem is derived from the observation that when the robot is a rigid object that can only move translationally, the configuration space is a convolution of the workspace and the robot. It makes use of the fast Fourier transform (FFT) algorithm to compute this convolution. The method is particularly promising for workspaces with many and/or complicated obstacles, or when the shape of the robot is not simple. This method can benefit from existing FFT experience and hardware.

Another basic problem in motion planning is collision detection and avoidance. Regarding collision detection, in [Kameyama92][16] a VLSI architecture is proposed, where each processor consists mainly of an arithmetic unit for 2D vector rotations and memory. In this distributed memory, a manipulator is represented by a set of discrete points covering its surface and obstacles by discrete volume elements (voxels) in workspace (see Figure 10). When the joint angles are specified, the coordinate transformation and the collision check can be performed in parallel without any interprocessor communication. Using 100 processors, each storing 50 manipulator points and $64 \times 64 \times 64$ voxels, the typical collision detection time becomes about 450 μs. Regarding collision avoidance, the problem is to modify a given path so that no collisions with obstacles or

16 A short version of this paper is [Kameyama91].

other robots occur. In [Fossa92], this problem is discussed for mobile robots. Unpredicted obstacles are avoided through the cooperation of the multiple navigation modules. Each module outputs a "fulfillment score" for each direction. The final motion direction is obtained by determining a point of maximum on the sum of scores for each direction. The modules are statically scheduled on different processors of a MIMD machine with a special purpose interconnection network.

Given a path generated by the motion planner, this path must be smoothed to be feasible for real robots. Robot trajectories can be optimized by genetic algorithms, providing the algorithm will consider the order and varying lengths of the trajectories. The trajectories consisting of a finite sequence of configurations are coded as genetic string. "Analogous cross-sites" are matched on the basis of similarity of discrete end-effector positions. In [Davidor90], the genetic algorithm optimizes the dynamic behavior of the robot, tracking a given trajectory. The performance shows improvements when compared with that of hill-climb and random search algorithms. In [Chan93], the genetic algorithm planes minimum-time trajectories. The planning procedure respects the limits on the torque values applied to the motor of each joint of the arm. Therefore, torque-bound information associated with the travelling from one configuration to the next is given for each configuration in the genetic string.

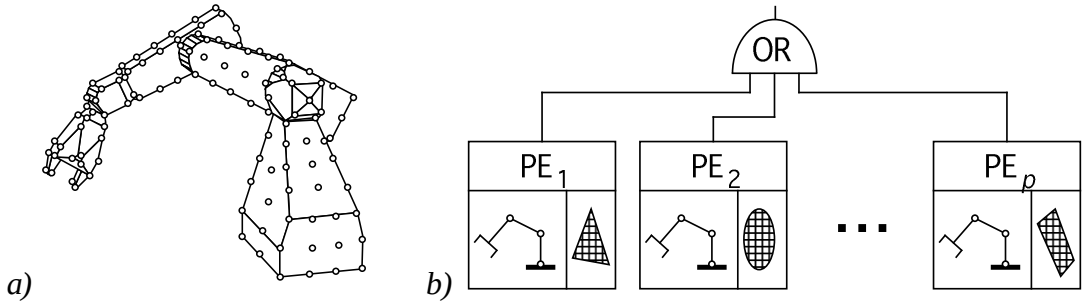


Figure 10: Example for parallel collision detection [Kameyama91]: a) A manipulator represented by a set of points covering its surface; b) An architecture, where each processor element (PE) consists of an arithmetic unit and memory for storing manipulator and obstacle information

A cellular automata-based path optimization for spray painting manipulators is given in [Hyotyniemi90]. The object surface is represented by single surface elements, which individually calculate the amount of "paint" they get. The processors operate on clusters of surface elements and gradually modify the trajectory by iterative refinement cycles. Unfortunately, the convergence cannot generally be guaranteed.

Another parallel path optimization technique for aerospace vehicles is presented in [Betts91]. The expense of trajectory optimization is split up into two factors: the cost of simulating a trajectory and the cost of computing gradient information for the optimization algorithm. The trajectory is broken into phases, which can be simulated in parallel, thereby reducing the cost of an individual trajectory. The non-linear optimization problem that results from this formulation produces a sparse Jacobian matrix. The Jacobian is computed using sparse finite differencing, which is also performed in parallel. The algorithm was implemented on the BBN-Butterfly GP1000. The results do not exploit sparsity and/or parallel processing in the non-linear programming algorithm itself. Finally, a

special case of path optimization is to find the shortest path in 3D space with polyhedral obstacles where the order in which the obstacles are encountered in this shortest path is known. Using *n* processors of a CREW PRAM, a solution can be found in O(log *n*) time [Bajaj86].

In [Voliotis92], the problem of path tracking in robotic manipulators' applications is studied. The path is generated as a sequence of elementary motions. The characteristic feature of the algorithm is that it avoids singularities, because there is no need to use inverse kinematics. Direction and proximity criteria are introduced. The algorithm is implemented on the Alliant FX/80.

4. Summary and Future Work

In Table 2, the references are classified in a 2D scheme, which is set up by the different approaches to motion planning and to parallel processing. 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.

Parallel approach	Graph-based approaches	Grid-based approaches	Potential field	Mathematical programming	Ancillary algorithms
Graph search	Shang92, Stifter93,	Barraquand91, Challou95, Qin96b			SchmidtBrauns91
Genetic algorithms	Chung91b, Shibata93,	Bessiere94, Solano93, Leung94, Pinchard95			Davidor90, Chan93
Cellular automata		Teng93,	Witkowski83, Won87, Shu90, Keymeulen91, Prassler95, Suzuki91, Dubash93, Yen93, Stiles94		Dehne89, Hyotyniemi90, Lozano Perez91
Neural nets		Lin91,	Huse90, Kalyayev93, Siemiatkowska93,	Lemmon91	
Processor farms	Gonzalez91		Solano94,		Bajaj86[17]
Static task scheduling	Rovetta92,		Fink91	Cela91	Betts91, Fossa92, Voliotis92
Special purpose hardware			Tarassenko91, Marshall93, Reid93, Ranganathan94,		Kameyama92, Kavraki93

Table 2: Classification of the references according their motion planning approach and parallelization approach[18]

[17] Actually, a CREW PRAM is used.

[18] Because some articles discuss more than one approach to motion planning or to parallelisation, the references may occur more than once. When an article covers multiple parts of a motion planning method, which are not all executed in parallel, then the reference is classified by the parallel processed part of the method.

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, such as coarse grids or line segments, were used. An interesting extension of grid-based approaches are polyhedral representations that may be used in order to plan a path on a lager scale of workspace and be combined hierarchically with the grid-based approach
- The number of DOFs was reduced such that the planning algorithm becomes less complex. For example, all the parallel potential field approaches work only in 2D. These approaches should be extended to higher-dimensional spaces so that they can be used for manipulators too. When using a terrain map, an interesting extension is the case of changing cost, i.e., searching in an *x*-*y*-*t*-map.
- No dynamic obstacles were regarded, thus, a C-space representation can be used in a simplified way. For example, in future work, genetic algorithms approaches could be extended by reactive planning for improving the motion despite large unexpected disturbances.

Additionally, further work could 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.

On the one hand, when applying the on-line motion planning in practice, today's methods do not fulfill the necessary time constraints. Adequate planning times are only possible by substantially reducing the time in one of the above ways. On the other hand, if motion planning can be performed in real-time, it can be included in closed loop in control algorithms. The concept of reactive planning then reduces to closed-loop real-time motion planning. Therefore, parallel real-time motion planning for complex robotics systems opens up new perspectives on some key issues in robotics such as motion planning in incompletely known or unknown environments, motion planning among fast moving obstacles, and multi-robot motion planning.

5. Acknowledgements

The work was performed at the Institute for Real-Time Computer Systems and Robotics, Prof. Dr. U. Rembold and Prof. Dr. R. Dillmann, University of Karlsruhe, D-76128 Karlsruhe, Germany. Many thanks to my colleagues Dr. Xiaoqing Cheng, Dr. Petra Bohner, and Michelle Specht for proofreading.

6. References

To each reference, a list of keywords is given. As far as possible, at least one keyword is chosen from each orthogonal keyword group. The keywords have the same order as in Table 1. As additional keywords, we use *introduction* and *survey*.

[Aggarwal88] Aggarwahl A., et al., "Parallel computational geometry". In: Algorithmica, vol 3, pp 293-327, 1988.
Keywords: Survey;

[Ahuactzin91] 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,
Keywords: Mobile robots; Manipulators; Cell decomposition; Genetic algorithms; MIMD computers; Torus networks; Scalable parallelism;

[Ahuactzin92] 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.
Keywords: Mobile robots; Manipulators; Cell decomposition; Genetic algorithms; MIMD computers; Torus networks; Scalable parallelism;

[Atallah89] 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.
Keywords: Skeleton; Shared memory systems;

[Bajaj86] Bajaj C., "An efficient parallel solution for Euclidean shortest path in three dimensions". In: Proc. 1986 IEEE Int. Conf. on Robotics and Automation, pp 1897-1900, 1986.
Keywords: Path optimisation; Shared memory;

[Barraquand89] 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.
Keywords: Manipulators; Multirobot environments; Potential field; Skeleton; Graph search; SIMD computers; Hypercube networks; Scalable parallelism;

[Barraquand91] 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.
Keywords: Potential field; Skeleton; Graph search; No parallel implementation; Scalable parallelism;

[Bessiere94] Bessière P., et al., "The Adriane's clew algorithm", In: Algorthmic Foundation of Robotics, Ed. by K. Goldberg, et al., A.K. Peters, 1994.
Keywords: Mobile robots; Manipulators; Cell decomposition; Genetic algorithms; MIMD computers; Torus networks; Scalable parallelism;

[Betts91] Betts J.T.; Huffman W.P., "Trajectory optimization on a parallel processor". In: Jour. of Guidance, Control, and Dynamics, vol 14 no 2, 1991, March-April, pp 431-9.
Keywords: Mobile robots; Path optimization; Static task scheduling; Shared memory systems; Specialist parallelism;

[Cela91] 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.
Keywords: Manipulators; Mathematical programming; Static task scheduling, No parallel implementation; Specialist parallelism;

[Challou93a] 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.
Keywords: Manipulators; Potential field; Skeleton; Graph search; MIMD computers; Hypercube networks;

[Challou93b] 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.
Keywords: Manipulators; Potential field; Skeleton; Graph search; MIMD computers; SIMD computers; Bus networks; Hypercube networks;

[Challou95] Challou D.J.; Boley D.; Gini M.; Kumar V., "A parallel formulation of informed randomized search for robot motion planning problems". In: Proceedings IEEE International Conference on Robotics and Automation, Nagoya, Aichi, Japan, May 21-27, 1995, pp 709-714.
Keywords: Manipulators; Potential field; Skeleton; Graph search; MIMD computers; SIMD computers; Bus networks; Hypercube networks;

[Chan93] Chan K.K.; Zalzala A.M.S., "Genetic-based minimum-time trajectory planning of articulated manipulators with torque constraints". In: IEE Colloquium on 'Genetic Algorithms for Control Systems Engineering' London, UK, 1993, 28 May, pp 4/1.
Keywords: Manipulators; Path optimization; Genetic algorithms; No parallel implementation; Scalable parallelism;

[Chung91a] 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.

Keywords: Mobile robots; Skeleton; Genetic algorithms; Neural nets; No parallel implementation; Scalable parallelism;

[Chung91b] 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.
Keywords: Mobile robots; Skeleton; Genetic algorithms; Neural nets; No parallel implementation; Scalable parallelism;

[Davidor90] Davidor Y., "Robot programming with a genetic algorithm". In: COMPEURO '90. Proceedings of the 1990 IEEE Int. Conf. on Computer Systems and Software Engineering, Tel-Aviv, Israel, 1990, 8-10 May, pp 186-91.
Keywords: Manipulators; Path optimization; Genetic algorithms; No parallel implementation; Scalable parallelism;

[Decuyper90] 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.
Keywords: Mobile robots; Potential field; Cellular automata; SIMD computers; Mesh networks; Scalable parallelism;

[Dehne89] Dehne F., Hassenklover A.-L., Sack J.-R., "Computing the configuration space for a robot on a mesh-of-processors". In: Parallel Computing, vol 12, 1989, pp 221-231.
Keywords: Mobile robots; C-space computation; Cellular automata; MIMD computers; Mesh networks; Scalable parallelism;

[Dubash93] 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.
Keywords: Multirobot environments; Mobile Robots; Potential field; Cellular automata; Mesh networks; Scalable parallelism;

[Fink91] 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.
Keywords: Manipulators; Cell decomposition; Potential field; Static task scheduling; MIMD computers; Specialist parallelism;

[Fossa92] Fossa M.; Grosso E.; Ferrari F.; Magrassi M.; Sandini G.; Zapendouski M., "A visually guided mobile robot acting in indoor environments". In: Proc. IEEE Ws on Applications of Computer Vision, Palm Springs, CA, USA, 1992, 30 Nov.-2 Dec., pp 180-308.
Keywords: Mobile Robots; Collision avoidance; MIMD computers; Special purpose networks; Specialist parallelism;

[Gonzalez91] Gonzalez J. S., Jones D.I., "An implementation on multiple transputers of a configuration space approach to robot obstacle avoidance". In T. S. Durrani, et al. (Eds.), Applications of Transputers 3. Proceedings of the Third Int. Conf. on Applications of Transputers, IOS Press-Amsterdam, pp 168-73, 1991.
Keywords: Manipulators; Cell decomposition; Processor farms; Static task scheduling; MIMD computers; Special purpose networks; Scalable parallelism;

[Gustafson88] Gustafson J. L., 1988, "Reevaluation Amdahl's law". In: Communications of the ACM, vol 31, no 5, pp 532-533.
Keywords: MIMD computers; SIMD computers;

[Huse90] 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.
Keywords: Mobile robots; Potenial fields; Neural networks; No parallel implementation; Scalable parallelism;

[Hwang92] Hwang Y. K., Ahuja N., "Gross motion planning – A survey". In: ACM Computing Surveys, vol 24, no 3, Sept 1992.
Keywords: Survey;

[Hyotyniemi90] Hyotyniemi H., "Locally controlled optimization of spray painting robot trajectories". In: Intelligent Motion Control. Proceedings of the IEEE International Workshop, 20.-22. Aug., 1990, Istambul, vol 1.1990 pp 283-287.
Keywords: Manipulators; Path optimization; Cellular automata; No parallel implementation; Scalable parallelism;

[Kalyayev93] 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.
Keywords: Mobile robots; Cell decomposition; Neural nets; Special purpose hardware; Mesh networks; Scalable parallelism;

[Kameyama91] Kameyama M.; Amada T.; Higuchi T., "Highly parallel collision detection VLSI processor for intelligent robots". In: 1991 Symposium on VLSI Circuits. Digest of Technical Papers, Oiso, Japan, 1991, 30 May-1 June, pp 500-6.
Keywords: Manipulators; Collision avoidance; Processor farms; Special purpose hardware; Scalable parallelism;

[Kameyama92] Kameyama M.; Amada T.; Higuchi T., "Highly parallel collision detection processor for intelligent robots". In: IEEE Jour. of Solid-State Circuits, vol 27, no 4, 1992, April, pp 500-6.
Keywords: Manipulators; Collision avoidance; Processor farms; Special purpose hardware; Scalable parallelism;

[Kavraki93] Kavraki L., "Computation of configuration-space obstacles using the fast Fourier transform". In: Proc. IEEE Int. Conf. on Robotics and Automation, Atlanta, GA, USA, 1993, 2-6 May, pp 255-61.
Keywords: C-space computation; Special purpose hardware; Scalable parallelism;

[Keymeulen91] 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.
Keywords: Mobile Robots; Potential field; Cellular automata; SIMD computers; Mesh networks; Scalable parallelism;

[Keymeulen94] 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.
Keywords: Mobile Robots; Potential field; Cellular automata; SIMD computers; Mesh networks; Scalable parallelism;

[Latombe91] Latombe J. C., "Robot motion planning", Kluwer Academic Publishers, 1991.
Keywords: Survey;

[Lemmon91] Lemmon M., "2-Degree-of-freedom Robot Path Planning using Cooperative Neural Fields". In: Neural Computation, vol 3, no 3 pp 350-362.
Keywords: Mobile robots; Mathematical programming; Neural nets; No parallel implementation; Scalable parallelism;

[Leung94] Leung C.H.; Zalzala A.M.S., "A genetic solution for the motion of wheeled robotic systems in dynamic environments". In: Int. Conf. on Control '94, Coventry, UK, 1994, 21-24 March, pp 760-4.
Keywords: Mobile Robots; Cell decomposition; Genetic algorithms; No parallel implementation; Scalable parallelism;

[Lin91] 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.
Keywords: Mobile Robots; Skeleton; Neural networks; Special purpose hardware; Mesh networks; Scalable parallelism;

[LozanoPerez91] Lozano-Pérez T., O'Donnell P. A., "Parallel robot motion planning". In: IEEE Int. Conf. on Robotics and Automation, Sacramento, California, April 1991, pp 1000-1007.
Keywords: Manipulators; C-space computation; Cellular automata; SIMD computers; Scalable parallelism;

[MacKenzie90] 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.
Keywords: Skeleton; Hypercube networks;

[Marshall93] 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.
Keywords: Mobile Robots; Potential field; Special purpose hardware; Mesh networks; Scalable parallelism;

[Mazer93] 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.
Keywords: Mobile Robots; Cell decomposition; Genetic algorithms; MIMD computers; Torus networks; Scalable parallelism;

[Paige85] Paige R. C., Kruskal C. P., "Parallel algorithms for shortest path problems". In: IEEE, 1985, pp 14-20.
Keywords: Graph search; Shared memory systems;

[Pinchard95] 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.
Keywords: Mobile robots; Skeleton; Genetic algorithms; No parallel implementation; Scalable parallelism;

[Prasanna92] 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.
Keywords: Survey;

[Prassler89] Prassler E.; "Electrical networks and a connectionist approach to path-finding". In: Proc. of the Int. Conf. "Connectionism in Perspective", Amsterdam, Elsevier, 1989.
Keywords: Mobile Robots; Potential field; Cellular automata; No parallel implementation; Mesh networks; Scalable parallelism;

[Prassler90] 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.
Keywords: Mobile Robots; Potential field; Cellular automata; No parallel implementation; Mesh networks; Scalable parallelism;

[Prassler94] 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.
Keywords: Mobile Robots; Potential field; Cellular automata; No parallel implementation; Mesh networks; Scalable parallelism;

[Prassler95] 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*.
Keywords: Mobile Robots; Potential field; Cellular automata; No parallel implementation; Mesh networks; Scalable parallelism;

[Qin96a] Qin C., Henrich D., 1996, "Randomized parallel motion planning for robot manipulators", Technical Report 5/96, Computer Science Department, University Karlsruhe. Keywords: Manipulators; Cell decomposition; Graph search; MIMD computers; Bus networks; Scalable parallelism;

[Qin96b] Qin C., Henrich D., 1996, "Path planning for industrial robot arms – A parallel randomized approach", In: Proceedings of the International Symposium on Intelligent Robotic Systems (SIRS'96), Lisbon, Portugal, July 22.-26., pp 65-72. Keywords: Manipulators; Cell decomposition; Graph search; MIMD computers; Bus networks; Scalable parallelism;

[Ranganathan94] 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.
Keywords: Mobile Robots; Potential field; Cellular automata; Special purpose hardware; Linear networks; Scalable parallelism;

[Reid93] 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.
Keywords: Mobile Robots; Potential field; Special purpose hardware;

[Rovetta92] 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.
Keywords: Manipulators; Skeleton; Static task scheduling; MIMD computers; Special purpose networks; Specialist parallelism;

[SchmidtBrauns91] 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.
Keywords: Mobile robots; Cell decomposition; Graph search; MIMD computers;

[Shang92] 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.
Keywords: Mobile Robots; Multirobot environments; Skeleton; Graph search; Shared memory systems; Scalable parallelism;

[Shibata93] 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.
Keywords: Mobile Robots; Mulirobot environments; Skeleton; Genetic algorithms; No parallel implementation; Skalable parallelism;

[Shu90] 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.
Keywords: Mobile Robots; Potential field; Cellular automata; SIMD computers; Mesh networks; Scalable parallelism;

[Siemiatkowska94] 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.
Keywords: Mobile robots; Cell decomposition; Neural networks; No parallel implementation; Scalable parallelism;

[Solano93] 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.
Keywords: Mobile robots; Manipulators; Skeleton; Genetic algorithms; No parallel implementation;

[Solano94] 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.
Keywords: Mobile Robots; Potential field; Processor farms; MIMD computers; Linear networks; Special purpose networks; Scalable parallelism;

[Stifter93] 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.
Keywords: Manipulators; Cell decomposition; Graph search; Shared memory systems;

[Stiles94] 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.
Abstract: Keywords: Mobile robots; Potential field; Cellular automata; SIMD computer; Mesh networks; Scalable parallelism;

[Suzuki91] 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.
Keywords: Mobile robots; Potential field; Cellular automata; No parallel implementation; Scalabale parallelism;

[Talbi92] 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.
Keywords: Mobile robots; Cell decomposition; Genetic algorithms, MIMD computers; Torus networks; Scalable parallelism;

[Talbi93] 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.

Keywords: Mobile Robots; Cell decomposition; Genetic algorithms; MIMD computers; Torus networks; Scalable parallelism;

[Tarassenko91] Tarassenko L., Blake A., "Analogue computation fo collison-free paths". In: IEEECROB, 1991, pp 540-545.
Keywords: Mobile robots; Potential field; Spcial purpose hardware; Mesh networks; Scalable parallelism,

[Teng93] 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.
Keywords: Multirobot environment; Cell decomposition; Cellular automata; SIMD computers; Hypercube networks; Scalable parallelism;

[Voliotis92] Voliotis S.D.; Christodoulou M.A., "Continuous path planning via a noninverting parallel algorithm". In: Robotica, vol 10, no pt.3, 1992, May-June, pp 205-16.
Keywords: Manipulators; Path optimization; Static task scheduling; MIMD computers; Bus networks; Specialist parallelism;

[Witkowski83] 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.
Keywords: Mobile robots; Potential field; Cellular automata; SIMD computer; Mesh networks; Scalable parallelism;

[Won87] 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.
Keywords: Potential field; Cellular automata; MIMD computers; Mesh networks; Hypercube networks; Scalable parallelism;

[Yen93] 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.
Keywords: Cell decomposition; Potential field; Cellular automata; No parallel implementation; Mesh networks; Hypercube networks; Scalable parallelism;