Dynamic Path Planning for a 7-DOF Robot Arm

Conference Paper · November 2006

DOI: 10.1109/IROS.2006.281798 · Source: IEEE Xplore

Proceedings of the 2006 IEEE/RSJ
International Conference on Intelligent Robots and Systems
October 9 - 15, 2006, Beijing, China

Dynamic Path Planning for a 7-DOF Robot Arm

Stefan Klanke, Dmitry Lebedev, Robert Haschke, Jochen Steil and Helge Ritter
Neuroinformatics Group
Faculty of Technology
University of Bielefeld
P.O.-Box 10 01 31, D-33501 Bielefeld, Germany
Email: {sklanke, dlebedev, rhaschke, jsteil, helge}@techfak.uni-bielefeld.de

***Abstract*— We present an on-line, robust, and efficient path planner for the redundant Mitsubishi PA-10 arm with 7 degrees of freedom (DOF) in non-stationary environments. Because of the specific kinematic model of the arm, path planning can be first reduced to a redundant 6-DOF problem in a 5D configuration space, which can be further decomposed into two problems: (i) 3D position planning in Cartesian space and (ii) planning in a 3D space composed of two orientation angles and an explicit parameterization of the arms redundancy. Position and orientation planning are interweaving and performed "on-the-fly" without explicit global knowledge of the environment using two instances of the dynamic wave expansion neural network (DWENN), an effective method for path generation in arbitrarily changing environments. The dynamic and explorative nature of the DWENN algorithm allows to treat stationary and dynamic obstacles in a unified manner. Through a number of simulative tests, we show that the planner is capable of reaching both a satisfactory robustness level and real-time performance, as required by many practical applications.**

I. Introduction

Path planning for robotic manipulators both in research environments and for industrial applications is a challenging problem, in particular if man-machine interaction in consequently highly dynamic environments is considered. In this context, the ultimate goal of the contemporary robotics research is to provide robotic manipulators with on-line, autonomous, robust, and, when possible, optimal or near-optimal path planning capabilities.

In contrast to numerous approaches to motion planning for mobile robots, many methods for robotic manipulators assume that the environment is stationary. For instance, [1] and [2] require an *off-line* precomputation of obstacle configurations before the path generation is performed. The approach in [3] generates subgoals to guide the planning in the discretized configuration space (C-space) of the robot, and exploits parallel processes for concurrent search for a valid path. [4] uses a genetic algorithm with a multi-criteria fitness function for the end-effector motion planning of a redundant manipulator.

There also exist several approaches to motion coordination of multiple manipulators. In [5], the time-velocity decomposition approach is transferred to motion planning of two manipulators. [6] and [7] consider cooperation of two manipulators in a shared workspace, but since the environment is highly-structured, the planning is reduced to a 2D case. A purely on-line approach to collision-free path planning and coordination of two robot arms is proposed in [8].

Among the most popular approaches are potential field methods existing in both numerical [9]–[11] and analytical form [12]–[15]. The well-known drawback of these methods is that they suffer from the local-minima-problem (which mainly concerns the analytical representation of potentials fields). Different prominent techniques for path planning in higher-dimensional configuration spaces are probabilistic (or randomized) methods [16]–[19]. They try to approximate the configuration space, i.e., to capture its connectivity using random sampling. Recently, however, also deterministic, incremental building of grids in higher-dimensional spaces has been considered as an alternative to path planning based on random sampling [20]. In addition, [21]–[23] discuss the heuristic grid-search and replanning algorithms which try to reduce the number of grid nodes to be processed during the path search. Note however that in a dynamic environment the probabilistic roadmap methods (even in their "light" version – with lazy collision checking) may fail, since the movement of obstacles will lead to elimination of roadmap edges, which may destruct the connectivity of the graph structure.

In this paper, we present a planner for a 7-DOF robot arm, which, thanks to its dynamic nature, can deal with virtually any type of environment, be it stationary, dynamic or unknown. Since path planning within a *high-dimensional* and *dynamic* configuration space suffers from the curse of dimensionality, we decompose the initial 7-DOF problem into simpler ones. Particularly, our planner orchestrates two 3D subplanners, which together solve a redundant 6-DOF in 5D problem. The remaining 7th DOF corresponds to the last roll joint of the robot arm and can be handled separately.

Each subplanner is an instance of the original dynamic wave expansion neural network (DWENN [24]), an efficient tool for path planning in time-varying and highly dynamic environments. The main feature of this grid based algorithm is that its "local" per-node complexity does not directly depend on the dimensionality of the configuration space.

Since path planning in each instance is done in an *on-line*, *explorative* fashion, all forbidden (obstacle) configurations of the arm are detected "on-the-fly" and are *dynamically* integrated into the path planning process. The global planner, which masters the two DWENN-based subplanners, exploits only two simple heuristics: (i) We control the overall number of steps assigned to each subplanner and (ii) we insert random steps for an already arrived subplanner, in case the other

1-4244-0259-X/06/$20.00 ©2006 IEEE

subplanner does not finish within a certain number of trials.

In this work, we report the results of thorough simulative tests, which reveal that in reasonably cluttered environments the planner is acceptably robust, and satisfies real-time requirements as needed by many real-world applications.

II. PA-10 Robot Arm Kinematics

The Mitsubishi PA-10 is a 7 axis general purpose robot arm. It is composed of a succession of roll joints (No. 1, 3, 5 and 7 in Fig. 1) and pitch joints (No. 2, 4 and 6). On the one hand the existence of a redundant DOF complicates the kinematics, but on the other hand can be exploited for obstacle avoidance and generation of smooth trajectories, if represented within the path planning algorithm. An elegant closed form solution for the inverse kinematics of redundant robot arms has been described in [25]. In the following, we give a brief summary tailored to the PA-10.

Starting from the base of the arm, conceptually each roll joint and the following pitch joint are combined to a spherical joint at locations 2, 4 and 6 in Fig. 1. Any given end-effector position and orientation determines the location of the last pitch joint 6. Since base joint 2 is fixed and the arm segments have fixed length, the spherical elbow joint 4 can only move on a circle around the axis connecting joint 2 and 6. Hence, the redundancy manifold is a circle in Cartesian space which can be parameterized by an angle α between 0 and 2π.

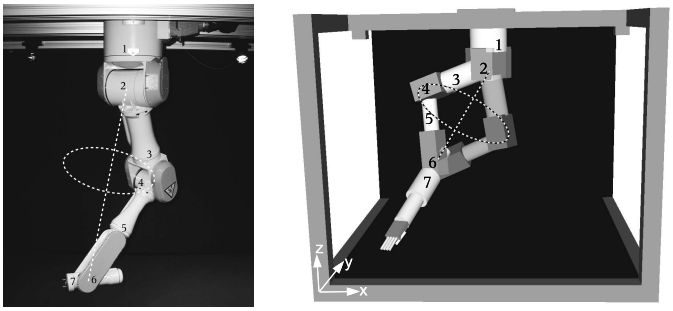


Fig. 1. Illustration of the PA-10 redundancy circle. The right picture shows two blended screen-shots from our simulation, including forearm and hand.

Due to inherent joint range limits some sectors on the redundancy circle yield an invalid arm posture. These forbidden sectors will be regarded as "internal" obstacles later and still can be handled by the formalism described in [25]. For any desired end-effector position and orientation we can freely choose the redundancy parameter α from a valid range, and then analytically calculate the corresponding joint angles.

The 7th arm joint controls the rotation of the end-effector about its roll axis, which can be predetermined without affecting the redundancy motion. Hence, we handle this joint separately, typically maintaining an upright orientation of the mounted robot hand during the whole movement. This effectively reduces the path planning to a redundant 6-DOF problem in 5D space, consisting of the Cartesian end-effector position as well as its azimuth and altitude angles relative to the table. Due to the nearly cylindrical shape of our robotic hand, this joint indeed can be neglected in collision checks.

III. Distributed Dynamic Path Planning

Since our approach aims at real time interaction of the robot with its environment, its planning system has to be capable of acquiring information about obstacles in an *on-line* and *dynamic* fashion. Additionally, planning takes place in a redundant 6D configuration space as described in the previous section. While up to date there are no real-time solutions available for such high-dimensional planning problems, which can dynamically integrate obstacle information, the dynamic wave expansion neural network has been proven to efficiently solve such problems in up to three dimensions [24]. This motivates to decompose the originally 6D planning problem into two 3D problems, which, however, interact in a highly non-trivial way because both 3D planners refer to a common real-world situation. This makes a planner's obstacle landscape dependent on the other planner's current configuration.

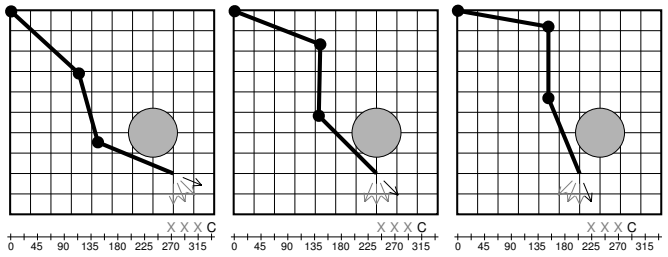


Fig. 2. The circular obstacle within the workspace induces different obstacle configurations in angular space (as marked by gray arrows and an X on the additional angular axis) depending on the actual end-effector position in Cartesian space. The current orientation of the arm is indicated by a black arrow and the letter C respectively.

Fig. 2 illustrates the effects of decomposition for a simpler 3-DOF arm. Here the 3D configuration space is split into a 2D positional and a 1D angular space, representing the Cartesian position and orientation of the end-effector respectively. The three shown configurations in positional space lead to different obstacle configurations if seen from angular space as illustrated by the gray arrows. The shift of the forbidden region in angular space becomes evident when looking at the additionally displayed discretized angular axis in Fig. 2, where forbidden angular configurations are marked with an X. Vice versa, forbidden end-effector positions in 2D Cartesian space change as a function of the end-effector orientation.

Note that we do not require an explicit representation of the obstacle landscape, neither within the subplanners nor within the overall planner. Rather, the planner actively explores its environment which is represented only implicitly within a continously updated virtual world model. The overall planner validates every aspired 7DOF arm posture computed from the current configurations of both subplanners employing collision checking routines within the world model. Because the obstacle landscapes of each subplanner strongly depends on the current configuration of the other planning instance, each planner has to reset and dynamically re-explore its forbidden configurations after a move of the other planner. An analytical computation of the obstacle configurations is possible, but computationally too costly.

A. Decomposition into Space and Orientation Planning

We adapt this decomposition principle to path planning for the PA10 within the remaining 6D configuration space. Specifically, we decompose the problem into two 3D subproblems governed by two independent subplanners: i) *position planning* of the end-effector in Cartesian space, and ii) planning of the end-effector's *orientation* as well as the *redundancy angle.* Hence, the configuration space of the first planner P_1 is the Cartesian position within the workspace, while that of the second planner P_2 is spanned by the two orientation angles azimuth and altitude as well as the angle α describing the position on the redundancy circle of the arm. Remember that we factored out the orientation of the end-effector about its roll axis, which can be handled independently by the 7th joint. Consequently, this leads to explicit redundancy control and multiple targets for the subplanner P_2, which can be easily handled by the DWENN algorithm.

B. Coordination and Exploration of Dynamic Obstacles

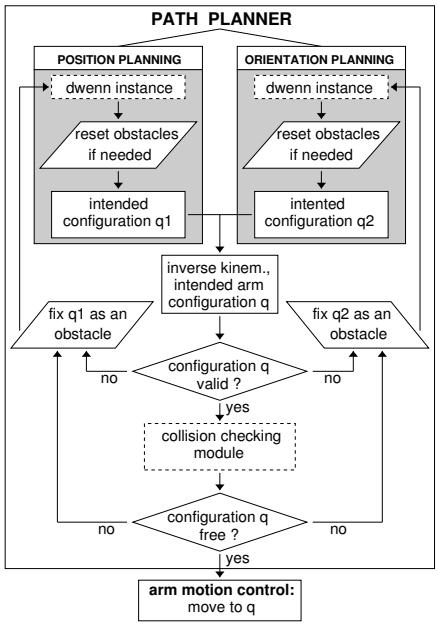


Fig. 3. Flow chart of the global path planning cycle.

The two subplanners are conceptually independent, use no common global information, and run alternating coordinated through a mastering planning process P as illustrated in Fig. 3. The mastering planner selects one of the subplanner instances P_i to execute the next few robot movements. Before an actual arm movement is performed, the intended new configuration of P_i is validated. To this end, the mastering planner combines the requested configuration of the active subplanner and the current configuration of the other subplanner and calculates a corresponding arm posture applying the inverse kinematics. If this posture cannot be found, violates joint angle limits, or would generate a collision as checked within the virtual world model, the requested 3D sub-configuration is marked as an obstacle within the active planner P_i. In this case, the subplanner either continues planning using the updated obstacle configuration, or waits for its next turn assigned by the master P. Otherwise the arm posture is actuated by the robot.

Due to the strong dependence of the obstacle landscape of one planning instance on the current configuration of the other planning instance, obviously *all* obstacles become dynamic from the point of view of each subplanner. Hence, each subplanner has to reset its acquired obstacle information, every time the other subplanner actually has moved the robot during its turn. Since the information about obstacles is collected "on-the-fly" in an explorative fashion, the planner can naturally deal with stationary, dynamic, and unknown environments.

C. Coordination Heuristics of the Mastering Planner

The mastering planner has to carefully assign appropriate planning chunks to each subplanner. On the on hand these chunks need to have an adequate length to enable the subplanners to acquire sufficient obstacle information, on the other hand these chunks have to coordinated, such that both planners simultaneously reach their respective 3D target.

If one subplanner P_a reaches its target significantly before the other planner P_b, the maneuverability of P_b is strongly restricted which may lead to a deadlock. Specifically, P_b may not reach its target(s), since all paths to the target(s) are blocked by obstacles.

To tackle this problem, we apply two heuristics in the mastering planner P: First, we estimate the remaining path length of both subplanners and from this calculate the probability to select a specific subplanner for the next few planning steps. Statistically, we thereby assign a number of local planning steps to each subplanner, which is proportional to its respective remaining path length. This *steps-scaling* forces the two planners to arrive at their targets as simultaneously as possible. Second, if nonetheless we arrive at the deadlock situation describe above, where planner P_b cannot reach its target anymore, we insert one random step for P_a after each N steps of P_b. In our experiments, we found $N = 15$ to be a suitable value.

IV. DWENN PATH PLANNING ALGORITHM

Recently, some research in robotics and AI was aimed at finding techniques for speeding-up and heuristic Al improvement of grid-search methods (their main difficulty being the exponential growth of complexity in the dimension of the configuration space). For example, [26] discusses how grid search can be optimized by heuristically reducing the number of neighbor expansions (depending on where the parent node is located in the neighborhood). [27]–[30] discuss incremental heuristic search algorithms which reduce the number of grid nodes to be recomputed by reusing the information from previous searches. Note that none of these algorithms can overcome the curse of dimensionality of the planning problem.

The DWENN algorithm is a grid-search technique, which was specifically designed for path planning in highly-dynamic environments. It is capable of generating dynamic distance potentials over the discretized configuration space of the robot.

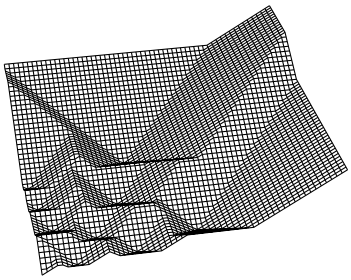


Fig. 4. Discretized potential field over a 60x60 workspace with 10 targets. The targets correspond to the minimum value points. In the shown case, the values of others grid nodes reflect the L_∞-distances to the nearest target.

Unlike the heuristic grid replanning techniques which aim at minimizing the number of expansion nodes, the main objective of the DWENN algorithm is to produce paths which do not contain oscillations and tend to be optimal even in an intensively changing environment.

The DWENN algorithm propagates waves of activity around one or multiple target configurations. At each instance of time a new wave emanates from these points and carries the information about the distance to the target, i.e., it propagates in such a way, that farther locations in the C-space accumulate larger activity values (see the illustration in Fig. 4). At each time step, the robot can move only to the neighboring location from which the activity has been inherited. The pseudo-code of a single iteration of the DWENN algorithm is shown in Fig. 5.

Two main features distinguish the DWENN algorithm from others. First, the update rule incorporates an *inhibitory* mechanism to reset the potential value of a node, if needed, and initiates propagation of inhibitory waves, which sequentially reset the activity of the nodes (lines 7, 19–21 in Fig. 5). This prohibits undesired path detours. Second, the local complexity of the DWENN algorithm does not depend on the dimensionality of the C-space (see [24], an example for a dynamic environment is presented in [8]).

Other features of the DWENN algorithm include:

- a *simple update rule*, which employs one addition and some binary checks only;
- the update rule determines the *selective* flow of the activity to the first "valid" neighbor (for which the GOOD_NEIGHBOR function in Fig. 5 returns true);
- the algorithm is *parameter-free*, i.e., there are no parameters, which may influence the efficiency of planning;
- the algorithm involves only integer-valued computation; its complexity grows linearly in the number of grid nodes;
- as reported in [24], DWENN qualitatively outperforms other existing algorithms (e.g., the algorithms in [31] and [32]).

```
00  t ← current time step;
01  for each grid node i do
02    if i = target then
03      act(i, t+1) ← 1; // targets have smallest activity
04    else if i = neighbor of a target
05      act(i, t+1) ← act(i, t) + 1;
06    else
07      act(i, t+1) ← 0; // inhibitory wave, if no "good" neighbor
08      for each neighbor j of node i,
                  starting with lastGood(i) do
09          if GOOD_NEIGHBOR(j, i, t) then
10            lastGood(i) ← j; // store path from i to j
11            act(i, t+1) ← act(j, t) + 2;
12            break; // simply take the first "good" neighbor
13  new configuration ← lastGood(current configuration)

14  bool GOOD_NEIGHBOR(j, i, t)
15    return
16      act(j, t) > 0 &&  // j is part of the activity wave
17      j ≠ obstacle &&  // j is not an obstacle
18      act(j, t) ≠ act(j, t−1) &&  // j carries new information
19      (act(i, t−1) + act(i, t) = 0 ||
20       act(j, t) < act(i, t))
21      // if i became inactive at t, last cond. is false for any j
22      // otherwise j should be closer to the target than i
```

Fig. 5. Single iteration of the DWENN path planning algorithm. Initially, all potential values $act(i, t = 0)$ are zero. Furthermore, it is assumed that the neighbors of node i are considered with respect to some preassigned ordering. It is also assumed that the nodes are expanded consecutively, i.e. first the target node(s), then its immediate neighbors are added, and so on.

V. Experimental Results

In our setup, a human-sized forearm and hand are mounted to the PA-10 as an end-effector, yielding a total length of 1915 mm from the base of the PA-10 to the fingertips. The additional DOFs of the hand are not taken into account in the planning process.

The PA-10 arm is mounted at the center of the ceiling of a nearly cubic workspace with an edge length of 1600 mm in x and y-direction. The height of the workspace (z-direction) is only 1350 mm. Note that the length of the arm and the size of the rigid end-effector impose serious difficulties for path planning even within an empty workspace.

For the following experiments, we used a 40x40x25 sized grid for the position planner (P_1), filling the lower 1000 mm of the workspace, and, thus, corresponding to a sampling interval of 40 mm along each axis. The combined orientation/redundancy planner (P_2) used a grid of size 30x40x30 (azimuth/altitude/redundancy).

A. Experiment 1: Reachability of Random Targets

In a first experiment, conducted to demonstrate the effectiveness of the proposed planning strategy, we randomly generated 10000 target postures from uniform sampling in the 3D Cartesian space and in the 2D space of azimuth and altitude orientation of the end-effector. Targets with no valid discretized solution were rejected. We then let the planner move to the desired target, always starting from a pre-defined "home" position (see Fig. 7). We measured the number of steps as well as the time the planner required to reach its target. If the planner did not arrive at the target within 500 steps, the attempt was canceled (which was the case for 12.2% of the

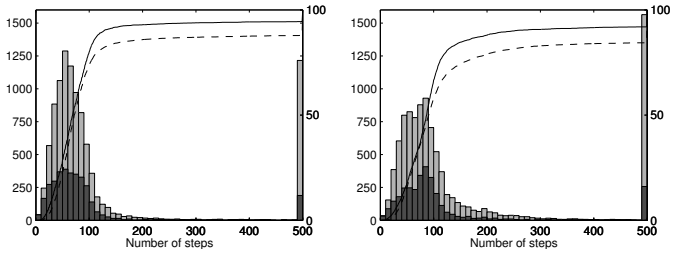
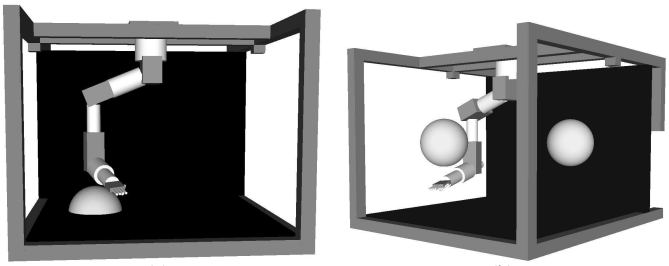


Fig. 6. Histogram and relative cumulative histogram showing the number of planning steps for reaching 10000 randomly selected targets. Left: Empty workspace. Right: With moving obstacle. The lighter bars and the dashed line apply to the full trial set, the darker bars and the solid line to the reduced trial set.



(a) (b)

Fig. 7. Home position for all experiments and location of spherical obstacles for Experiment 2 and 1b.

trials in an empty workspace). For the accomplished targets, the planner took 76 steps or 18.2 seconds on average to finish. The distribution of planning steps is illustrated in Fig. 6. If we reduce the set to those end-effector targets which have 15 feasible (out of 30 possible) redundant solutions, only 3399 targets remain and the error rate drops to 5.6%. This result emphasizes the hardness of the problem as well as the ability of the planner to exploit the redundancy manifold.

To demonstrate the effect of the proposed heuristics employed in the coordinating planner, we repeated the experiment without the heuristics, i.e. just with alternating steps of the subplanners. This led to a failure rate of 14.4%, resp. 7.7% for the reduced target set.

To demonstrate the ability to handle dynamically moving obstacles, we repeated the experiment moving a spherical obstacle up and down in the workspace at a velocity of 4cm/s (the left sphere in Fig. 7b). This resulted in a slightly increased error rate of 15.6% resp. 8.1% for the reduced target set.

In real applications, one must somehow handle unsuccessful planning attempts. For this, we suggest to define workspace-specific "safe" positions as intermediate targets and to repeat the planning process from different starting points.

B. Experiment 2: Reachability of the Table Surface

In this experiment, we demonstrate the capabilities of our path planning system in a task which resembles approaching movements in the context of grasping, and allows to easily visualize the results. To this aim, we created a set of targets by specifying a regular grid of 2D positions on the table (bottom of the workspace). The target height was fixed to z=50 mm above the table. To generate target orientations, we sampled the azimuth and altitude angles of the end-effector orientation, yielding up to 97 possible targets for each position on the table.

We then measured (1) how many of the targets are theoretically reachable, i.e. there is at least one valid arm posture, and (2) how many of these tasks the planner accomplished. Again, each trial was started from the home position. In a second and third series, we placed obstacles inside the workspace (the hemisphere and the two spheres in Fig. 7a and b, respectively).

Fig. 8 illustrates the results of this experiment. Note that our planner reaches all positions on the table nearly equally well. Only in the lower right corner the performance decreases slightly, which is due to violation of joint limits.

In the third series (bottom row of Fig. 8) the two spherical obstacles severely restrict the maneuverability of the arm. Still, averaged across all table points, our planner manages to reach 85% of the theoretically possible target postures.

VI. Conclusion

We presented a real-time path planner for the 7-DOF Mitsubishi PA-10 arm. The planner combines two 3D path planning instances responsible for the end-effector's position and orientation, respectively. We apply two heuristics which control the overall number of steps for each subplanner and initiate random steps in case one subplanner has arrived, while the other has not. Path planning is performed using the DWENN algorithm, designed specifically for path generation in highly dynamic environments. Path generation in each subplanner is performed on-line, and the obstacle information is collected "on-the-fly", which allows the planner to deal potentially with dynamic and unknown obstacles. The planner has been thoroughly evaluated in a number of simulative scenarios, including random target configurations, as well as configurations at the table surface, which is especially important in the context of grasping. These tests reveal that our planner is reliable and fast enough for real-world applications.

References

[1] Y. Ting, W. I. Lei, and H. C. Jar, "A path planning algorithm for industrial robots," *Computers and Industrial Engineering*, vol. 42, pp. 299–308, 2002.

[2] M. Tarokh, "Fast path planning for robot manipulators by formation-posture decomposition," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'95)*, 1995, pp. 138–143.

[3] C. Qin and D. Henrich, "Path planning for industrial robot arms - a parallel randomized approach," in *Proc. Int. Symp. on Intelligent Robotic Systems (SIRS'96)*, 1996, pp. 65–72.

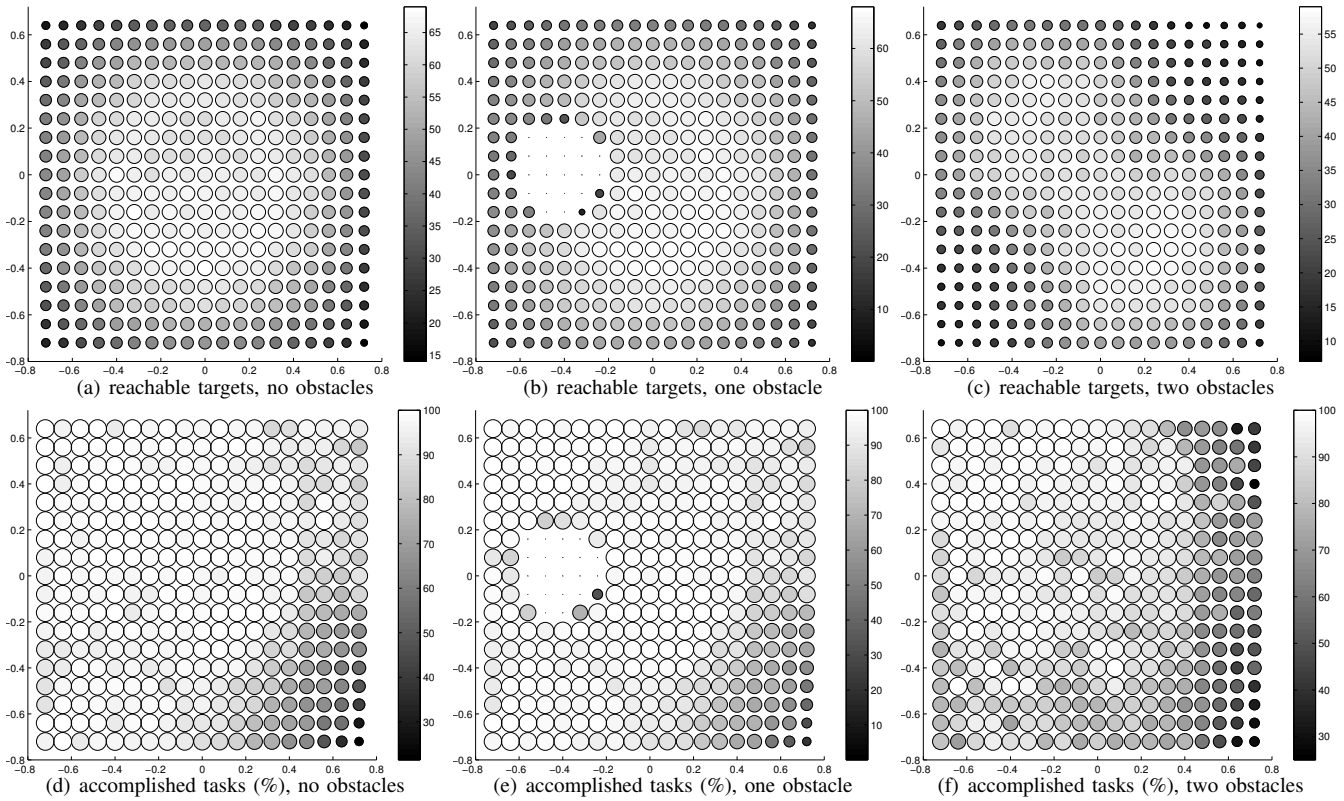
[4] L. Tian and C. L. Collins, "Motion planning for redundant manipulators using a floating point genetic algorithm," *Journal of Intelligent and Robotic Systems*, vol. 38, pp. 297–312, 2003.

[5] M. A. Ridao, E. F. Camacho, J. Riquelme, and M. Toro, "An evolutionary and local search algorithm for planning two manipulators motion," *Journal of Robotic Systems*, vol. 18, pp. 463–476, 2001.

[6] T. Li and J.-C. Latombe, "On-line manipulation planning for two robot arms in a dynamic environment," *International Journal of Robotics Research*, vol. 16, pp. 144–167, 1997.

[7] X. Cheng, "On-line collision-free path planning for service and assembly tasks by a two-arm robot," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA'95)*, 1995, pp. 1523–1528.

[8] D. V. Lebedev, J. J. Steil, and H. J. Ritter, "An on-line neural network based approach to dynamic path planning and coordination of two robot arms," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'05)*, 2005.



(a) reachable targets, no obstacles (b) reachable targets, one obstacle (c) reachable targets, two obstacles

(d) accomplished tasks (%), no obstacles (e) accomplished tasks (%), one obstacle (f) accomplished tasks (%), two obstacles

Fig. 8. Top row: Number of reachable targets (forearm directions) per table position. Bottom row: Relative number of accomplished tasks ($\hat{=}$ planner performance). Left column: empty workspace; middle column: one obstacle on the table (Fig. 6a); right column: two obstacles (Fig. 6b). All numbers are coded by both the size and the gray tone of a circle at the corresponding position on the table. The PA-10 is mounted at x, y-coordinate (0,0).

[9] J. Barraquand, B. Langlois, and J.-C. Latombe, "Numerical potential field techniques for robot path planning," *IEEE Trans. Syst., Man, Cybern.*, vol. 22, pp. 224–241, 1992.

[10] G. Bugmann, J. G. Taylor, and M. Denham, "Route finding by neural nets," in *Neural Networks*, J. G. Taylor, Ed., 1995, pp. 217–230.

[11] A. A. Kassim and B. V. K. Vijaya Kumar, "Path planning for autonomous robots using neural networks," *Journal of Intelligent Systems*, vol. 7, pp. 33–56, 1997.

[12] O. Khatib, "Real time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, pp. 90–99, 1986.

[13] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 501–518, 1992.

[14] Z. X. Li and T. D. Bui, "Robot path planning using fluid model," *Journal of Intelligent and Robotic Systems*, vol. 21, pp. 29–50, 1998.

[15] Y. Wang and G. S. Chirikjian, "A new potential field method for robot path planning," in *Proc. of IEEE Int. Conf. on Robotics and Automation (ICRA'00)*, 2000, pp. 977–982.

[16] N. M. Amato and Y. Wu, "A randomized roadmap method for path and manipulation planning," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA'96)*, 1996, pp. 113–120.

[17] R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2000, pp. 521–528.

[18] G. Sánchez and J.-C. Latombe, "Using a PRM planner to compare centralized and decoupled planning for multi-robot systems," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2002, pp. 2112–2119.

[19] G. Song, S. Thomas, and N. M. Amato, "A general framework for PRM motion planning," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA'03)*, 2003, pp. 4445–4450.

[20] S. M. LaValle, M. S. Branicky, and S. R. Lindemann, "On the relationship between classical grid search and probabilistic roadmaps," *International Journal of Robotics Research*, vol. 23, pp. 673–692, 2004.

[21] M. Likhachev, G. J. Gordon, and S. Thrun, "ARA*: Anytime A* with provable bounds on sub-optimality," in *Advances in Neural Information Processing Systems 16*, 2003.

[22] A. Stentz, "The focussed D* algorithm for real-time replanning," in *Proc. Fourteenth Int. Joint Conf. on Artificial Intelligence*, 1995, pp. 1652–1659.

[23] K. I. Trovato and L. Dorst, "Differential A*," *IEEE Trans. Knowledge Data Eng.*, vol. 14, pp. 1218–1229, 2002.

[24] D. V. Lebedev, J. J. Steil, and H. J. Ritter, "The dynamic wave expansion neural network model for robot motion planning in time-varying environments," *Neural Networks*, vol. 18, pp. 267–285, 2005.

[25] P. Dahm and F. Joublin, "Closed form solution for the inverse kinematics of a redundant robot arm," *Internal Report IRINI 97-08*, 1997.

[26] J. Kuffner, "Efficient optimal search of Euclidean-cost grids and lattices," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'04)*, 2004.

[27] S. Koenig, M. Likhachev, Y. Liu, and D. Furcy, "Incremental heuristic search in artificial intelligence," *Artificial Intelligence Magazine*, vol. 155, pp. 93–146, 2004.

[28] D. Ferguson and A. Stentz, "The delayed D* algorithm for efficient path replanning," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA'05)*, 2005.

[29] D. Ferguson, M. Likhachev, and A. Stentz, "A guide to heuristic-based path planning," in *Proc. of the Workshop on Planning under Uncertainty for Autonomous Systems, Int. Conf. on Automated Planning and Scheduling (ICAPS'05)*, 2005.

[30] M. Likhachev, D. Ferguson, G. Gordon, A. Stentz, and S. Thrun, "Anytime dynamic A*: An anytime, replanning algorithm," in *Proc. of the Int. Conf. on Automated Planning and Scheduling (ICAPS'05)*, 2005.

[31] R. Glasius, A. Komoda, and S. C. A. M. Gielen, "Neural network dynamics for path planning and obstacle avoidance," *Neural Networks*, vol. 8, pp. 125–133, 1995.

[32] S. X. Yang and M. Meng, "Neural network approaches to dynamic collision-free trajectory generation," *IEEE Trans. Syst., Man, Cybern. C*, vol. 31, pp. 302–318, 2001.