

Multiresolution Path Planning in Dynamic Environments for the Standard Platform League

Ricarda Steffens, Matthias Nieuwenhuisen, and Sven Behnke

*Autonomous Intelligent Systems Group, Institute for Computer Science VI,
University of Bonn, Germany*
{steffens, behnke}@cs.uni-bonn.de, nieuwenh@ais.uni-bonn.de

Abstract—Following obstacle free paths towards the ball and avoiding opponents while dribbling are key skills to win soccer games. These tasks are challenging as the robot’s environment in soccer games is highly dynamic. Thus, exact plans will likely become invalid in the future and continuous replanning is necessary. The robots of the RoboCup Standard Platform League are equipped with limited computational resources, but have to perform many parallel tasks with real-time requirements. Consequently, path planning algorithms have to be fast.

In this paper, we compare two approaches to reduce the planning time by using a multiresolutional representation or a log-polar representation of the environment. Both approaches combine a detailed representation of the vicinity of the robot with a reasonably short planning time. We evaluated both methods in different simulated and real-robot tests and compared them to a uniform grid. Our experiments show that, while multiresolutional approaches provide paths of similar quality, the computational efficiency increases immensely.

I. INTRODUCTION

A basic skill for autonomous robots is the ability to plan collision-free paths. In the humanoid soccer domain, a common approach is to determine a gait target vector, which controls the direction and velocity of the robot’s motion, by incorporating the target position and the position of obstacles. Hence, perceptions are directly mapped to actions. The mapping may depend on additional factors, like the role assigned to the robot or the game state, but it does not consider the (foreseeable) future. This can lead to inefficient obstacle avoidance, e. g., the robot passes an obstacle on the side closer to its target just to be blocked by the next obstacle in the same direction. Fig. 1 shows a typical game situation.

On mobile robot platforms, the computational resources are often limited due to weight and power constraints. Accordingly, exact planning even of relatively small problem instances is not possible in real-time. Moreover, performing time-consuming planning and committing to a long-term plan is no option in highly dynamic domains like soccer. Because of the limited capabilities of the robot’s sensors, it is not possible to estimate precise obstacle positions. Therefore, the environment is not only dynamic, but path planning has to deal with uncertainties.

Thus, we propose to use approximate path planning with replanning every time a new state of the environment is perceived. With increasing time since the last perception, the prediction of the world state becomes more uncertain. Consequently, planning steps in the far future should be more

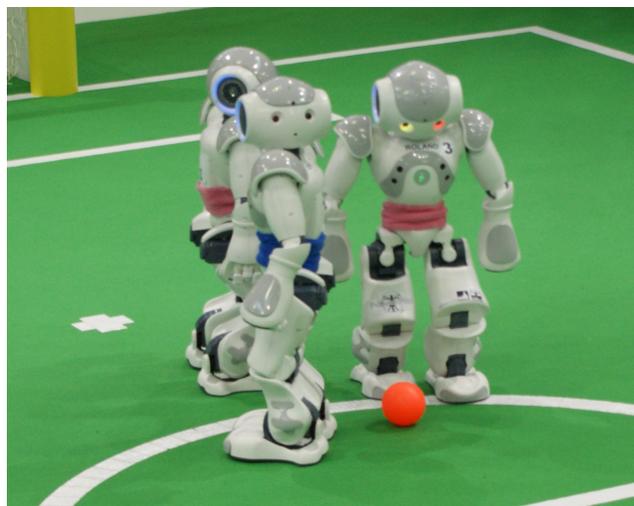


Fig. 1. Reactive obstacle avoidance can produce dribbling trajectories close to opponent field players. Prior planning can reduce the risk of losing ball possession.

approximative than planning steps that have to be executed immediately, as the former will likely be invalid at the time they are executed. In order to reduce the complexity of the plan representations, we employ multiresolutional approaches, namely, a local multiresolutional grid and a log-polar grid. In both representations, the resolution decreases with increasing distance to the robot.

After discussing work related to ours, we describe the robot hardware in Sec. III. In Sec. IV we detail the representations that we compare. We present simulated and real-robot experiments that we performed to evaluate the representations in Sec.V.

II. RELATED WORK

Our humanoid soccer team NimRo uses a hierarchical reactive approach to control the robot’s motion [1]. In contrast, our novel approach is based on planning. It considers the foreseeable future to determine obstacle-free robot paths. Continuous replanning allows for always considering the most recent sensory information and for quickly reacting to changes in the environment.

Many planning-based systems exist in the literature. The key challenge is the computational complexity of real-time

planning and execution. Kaelbling and Lozano-Pérez reduce the complexity of task planning by top-down hierarchical planning [2]. In their approach, an agent commits to a high-level plan. The refinement of abstract actions is performed at the moment an agent reaches them during plan execution. We follow the same assumption that there is likely a valid refinement at the time an abstract action has to be concretized, and that every plan can be reversed without huge costs in case that there is no such refinement.

A method for resource-saving path planning is the local multiresolution Cartesian grid [3]. It employs multiple robot-centered grids with different resolutions, and nests them hierarchically while connecting them through adjacencies. This representation resembles our approach to approximate plan steps with increasing distance to the robot. In addition, it was designed for soccer robots and, consequently, considers the present circumstances.

Apart from Cartesian occupancy grid maps [4], polar coordinate based grids can be found for egocentric robot motion planning in the literature. In this approach, the environment close to the robot has a high Cartesian resolution that decreases with the distance, due to the fixed angular resolution. Polar grids with hyperbolic distance functions are used to represent infinite radii within a finite number of grid cells. This property is used to plan long-distance paths in outdoor environments [5].

Another kind of polar grids are log-polar grids [6]. Like the local multiresolution grid, this approach emphasizes a more precise path planning in the robot's vicinity. Furthermore, polar grids have the advantage of an easy integration of obstacles perceived by ultrasonic sensors and cameras.

III. ROBOT PLATFORM

In the RoboCup Standard Platform League (SPL), Nao robots from Aldebaran Robotics are used [7]. The Nao V3+ edition, used in recent RoboCup competitions, has 21 degrees of freedom. The environment is perceived by two cameras, from which only one can be used at a time, and two ultrasonic sensors. In our system, we continuously switch between the two cameras, which results in a frame rate of approx. 14 Hz. The ultrasonic sensors are located at the robot's chest covering an angle of approx. 110° in front of the robot. The detection range is from 300 mm to 700 mm [8].

The ultrasonic sensors measure the distance towards an obstacle, but not its precise angular position. In contrast, both cameras provide the exact direction towards an obstacle, but no precise distance. This leads to uncertainties which have to be taken into account.

The Nao robot is equipped with a x86 AMD Geode LX 800 CPU running at 500 MHz. It has 256 MB of RAM and 2 GB of persistent flash memory [9]. The built-in processor has the advantage of low power consumption, with the tradeoff of low computational power. Compared to state-of-the-art computer systems, the resources are rather limited. Hence, a low system load is an important requirement for the development of new software components.

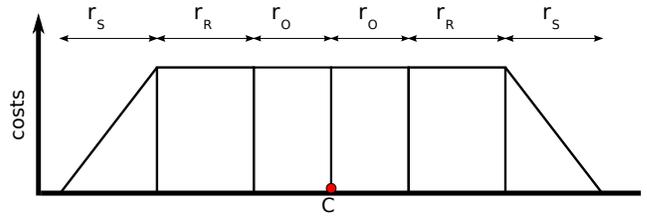


Fig. 3. Obstacles C are modeled as their own radius r_O and the radius of the robot r_R with maximum costs and a safety margin r_S with decreasing costs.

Our software is based on the framework of the German SPL-team B-Human [10]. The framework consists of several modules executing different tasks. In addition to modules for, e.g., perception or behavior control, a simulator called SimRobot is provided. For our tests, we integrated a new path planning module into the framework.

IV. PATH PLANNING REPRESENTATIONS

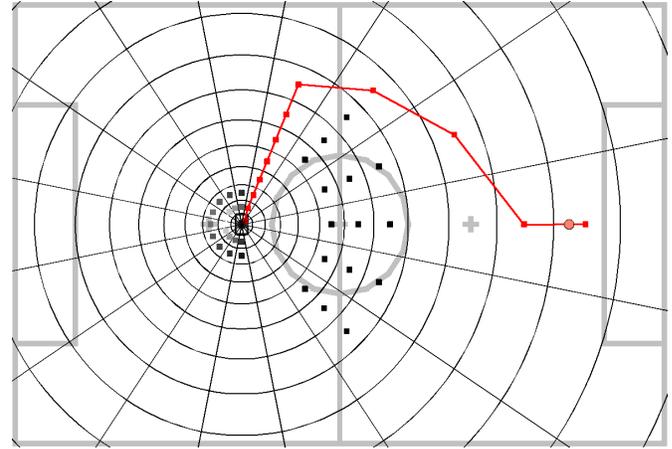
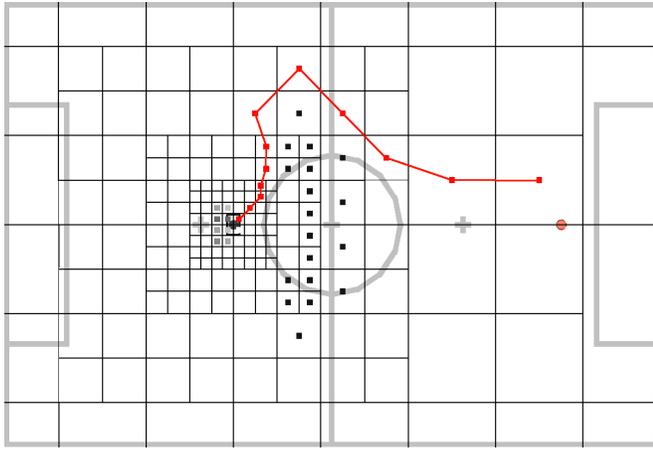
At RoboCup 2010, we used reactive target selection and obstacle avoidance behaviors. Thus, a gait target vector (v_x, v_y, v_θ) , which determines the walking speed in forward, lateral, and rotational direction, respectively, is determined merely by direct perceptions and active behaviors. Typical behaviors are *go_to_ball* or *avoid_obstacle*. Going to the ball leads to a gait target vector towards the estimated ball position. Obstacles can be seen as repulsive forces affecting the direction of the gait target vector in this notion.

Our approach to planning a path to a target introduces a planning layer between abstract behaviors and motion control. Waypoints provided by this layer are used to determine the needed velocities. The abstract behaviors configure the planner and perceptions are integrated into the planner's world representation.

In our implementation, we use the A*-algorithm with a closed list. We implemented our planner with three different types of representations: a uniform, a local multiresolution and a log-polar grid. All representations have in common that their coordinate system is egocentric, i.e., they are translated and rotated according to the robot's pose. In the following section, we describe our uniform grid representation and detail the two non-uniform grid representations.

A. Uniform Grid

A commonly used representation of the environment are uniform grid maps. These discretize the environment into equally-sized cells. Cells marked as occupied correspond to obstacles. The perceived obstacles are initialized with their own radius. The approximate robot radius and a safety margin with linear decreasing costs are added to the obstacle, before it is inserted into the map. This simplifies the planning problem to finding a path for a robot that is reduced to a point and allows to plan paths through robots standing close to each other, but avoids the vicinity of obstacles if this is possible without huge costs. Our obstacle model is depicted in Fig. 3.



(a) Local multiresolution grid with five levels and 8×8 cells on each level.

(b) Log-polar grid with 16 discrete steps for angle and distance respectively.

Fig. 2. Non-uniform grids allow to cover a given area with multiple orders of magnitude less cells than uniform ones. Depicted are planned paths on a soccer field (red dots and lines) starting from the robot position in the center of the grid to the ball 3 m in front of it. The robot faces towards the positive x -axis. Occupied cells are marked by black dots.

The center of a grid cell can be interpreted as a node of a graph with edges to the eight neighboring cell centers. Therefore, it is possible to use standard graph searches as the A*-algorithm (cf. [11]).

Due to the simple structure of the grid, there is no need to explicitly model the cell connectivity. Furthermore, the cost calculation per step is reduced to two cases: Diagonal or straight steps. Because of the simplicity, it can be easily implemented, and arbitrary environments can be represented equally accurate, up to a freely chosen resolution. This makes uniform grids a good choice for many applications. A major disadvantage is the computational complexity, which does not scale well with increasing grid size. Especially in environments with special characteristics, like a soccer field, more efficient representations are possible.

B. Local Multiresolution Grid

An efficient path planning algorithm is local multiresolution path planning [3]. Besides the computational advantages of multiresolution, the uncertainty of local sensing and of the own and the opponent's movements are implicitly taken into account with an increasing cell size.

The grid consists of multiple robot-centered grids with different resolutions embedded into each other. With increasing distance to the robot, the grid resolution decreases. This models the uncertainties caused by local sensing with only relative precision and by the dynamic environment. Local multiresolution planning utilizes the fact that the world changes continuously while the agents of the own and the opponent team move. Hence, it is not worthwhile to make detailed plans for the far-future.

More formally, the environment is discretized into a square $M \times M$ grid. Recursively, a grid is embedded into the inner part $\left[\frac{M}{4} : \frac{3M}{4}\right] \times \left[\frac{M}{4} : \frac{3M}{4}\right]$ of the grid at the next coarser level. The cell area of the inner grid is a quarter of the cell area of the outer grid. In order to cover the same area as

a uniform $N \times N$ grid, only $(\log_2(N/M) + 1) M^2$ cells are necessary.

Fig. 2a shows an 8×8 local multiresolution grid with a minimum cell size of 10 cm that covers an area of 163.84 m^2 with 256 cells using five grid levels. In contrast, a uniform grid covering the same area consists of 16,384 cells.

The neighborhood of an inner grid cell consists of eight neighbors, similar to the neighborhood of the uniform grid. However, the cells at the border to the lower-level grid have seven neighbors at the edges and six neighbors at the corner. Likewise, the grid cells at the border to the higher-level grid have nine neighbors and eight neighbors, respectively. In our implementation, the connections between two neighbors are encoded as edges of a graph.

Incorporating obstacles into the grid is analog to the uniform grid. But, since the cell size is not uniform, an obstacle in the lower hierarchy levels may be completely covered by just one cell or by more than one if it is close to the cell boundary. In this case, the costs of this obstacle are distributed over the covered cells. Furthermore, the costs of obstacles in larger cells are reduced because, presumably, not the complete cell is blocked by it and for this reason the cell is still partially traversable.

For path planning with the A*-algorithm, the costs of each step are calculated by means of the Euclidean distance of the centers of both cells and the added costs of the target cell. The employed heuristic is the distance from the current grid cell to the target.

The advantages of this representation are the low requirements on the robot's memory and on computational power. Moreover, uncertainties occurring in dynamic environments are considered by the increasing cell size with increasing distance from the robot.

C. Log-Polar Grid

Although our soccer robots are able to walk omnidirectionally, the best speed can be achieved in forward direction. Furthermore, the robot's sensing capabilities are designed for perceiving the environment in front of the robot. Accordingly, walking in forward direction towards the target, i.e., to the next waypoint of the plan, is preferred for long distances. On a soccer field with only smaller obstacles, it is likely that relatively long straight segments are often part of the plans. Because of our grid representations being egocentric, a target in front of the robot is on the positive x-axis. As this axis is a cell boundary in every resolution, this case is not well supported by the local multiresolution grid. Due to imprecise measurements of the target and inaccurate motion execution, distant targets in a uniform grid presumably change their respective cells, too.

This leads us to a grid representation that fits the robot's motion and sensing characteristics in a more efficient way. In contrast to the Cartesian grid representations, a polar grid representation provides a straight path towards targets in front of the robot if there are no obstacles. Additionally, sensory information relevant to path planning is initially provided in polar coordinates on our robots. Ultrasonic measurements are represented as a distance and an apex angle, and visual perceptions are estimated distances and directions to obstacles. Both can be easily incorporated in a grid representation in polar coordinates.

In polar coordinates, the environment is represented as an angle θ and a distance ρ with regard to the robot's pose, written as a tuple (θ, ρ) . In our coordinate system, the robot faces in the direction of the positive x-axis. We discretize the angular component θ into T equally sized partitions. The first partition is chosen in a way that the positive x-axis becomes the angle bisector of the angle represented by this partition. Hence, small angular inaccuracies in the perception or gait execution will not change the grid cell of a waypoint or the target.

To reach the implicit consideration of uncertainties and computational advantages of the multiresolutional grid, the cells of our polar grid grow exponentially with the distance. In order to achieve this, the logarithm of the distance to the robot is partitioned. To define a minimal cell size and still achieve a reasonable growth until the maximum radius, we use a slightly shifted logarithm to avoid the initial strong slope of the logarithm. The calculations to determine the polar coordinates (θ, ρ) of a point (x, y) in Cartesian coordinates, and the corresponding discretized grid cell (r, t) are

$$\begin{aligned}\rho &= \log_b \left(\left(\frac{\sqrt{x^2 + y^2}(b-1)}{l} \right) + 1 \right), \\ \theta &= \arctan\left(\frac{y}{x}\right), \\ (r, t) &= \left(\lfloor \rho \rfloor, \lfloor \left(\frac{T}{2\pi}\right)\theta + 0.5 \rfloor \right),\end{aligned}$$

where b is the base of the logarithm, l is the minimal cell size and T the number of angular partitions.

The inverse operation therefore is described by

$$(x, y) = \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix} \left(\frac{(b^{r+0.5} - 1) * l}{b - 1} \right).$$

In our implementation, we use a base of $b = 1.1789$ and a minimal cell size of $l = 100 \text{ mm}$. With 16 cell rings, we reach a maximum radius of 7211 mm , which is sufficient to plan paths for any two points within the SPL field boundaries. We use 16 steps for the angular component as well, leading to 256 cells in total, the same number of grid cells used in the local multiresolution grid. The resulting grid is depicted in Fig. 2b.

In the polar grid representation, the obstacles are treated analogously to the local multiresolution grid.

The costs of each step are calculated by means of the Euclidean distance of the centers of both cells, as in the local multiresolution grid. The heuristic is computed likewise. The cell distances are precalculated to decrease the computational complexity. Thus, single node expansions of a planner are not more costly in this representation than in uniform grids.

D. Implementation Details

In our 2D planning implementation, we neglect the orientation and velocities of the robot for efficiency reasons. Accordingly, due to the fast replanning, sudden changes of the gait target vector are possible. To avoid this, we introduce a virtual obstacle behind the robot, which represents its starting speed (Fig. 4). The polar grid representation employs a half circle with cost interpolation between a minimum at the edges and a maximum at the midway of the circle segment. In contrast, the Cartesian grid representations use a rectangle having the same characteristic. When the robot is moving, this obstacle is opposed to the gait target vector with costs corresponding to the scalar value of the velocity.

To generate motion commands for a planned path, the planning module sends the next waypoint on the path to the motion control in every execution cycle. Replanning is performed if the robot's movement exceeds a threshold. Between planning calls, the waypoint is adjusted using odometry data. The resulting gait target vector is a weighted sum of position of and angle to the waypoint relative to the robot.

V. EVALUATION

We evaluated our planning algorithms in simulation and on a Nao robot. In order to evaluate the quality of the different approaches, we investigated their computing time as well as the efficiency of the planned paths.

A. Planning Time & Node Expansions

The computing time of the three planners was measured in the simulator and on a Nao robot with regard to four different test cases, representing all possible constellations on the soccer field. Those are, 1. that no obstacle is detected, 2. obstacles are detected either in the ultrasonic sensor measurements or 3. through cameras, and 4. obstacles are captured by both, ultrasonic sensors and cameras. In every test case, the target was

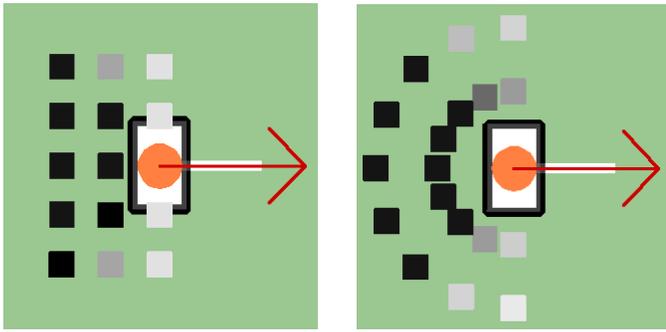


Fig. 4. An artificial obstacle leading to preference of paths towards the robot's current walking direction to avoid sudden directional changes. Cells with higher costs are darker. The red arrow is the gait target vector (left: Cartesian grid, right: polar grid).



Fig. 5. Field configuration used in the timing tests with camera support. The planning is performed on the red circled robot. The orange ball marks the target.

3000 mm in front of the robot and, according to each test case, the obstacle positions differ. The field configuration for test case four, in which obstacles are detected through ultrasonic sensors and cameras, is depicted in Fig. 5. In order to avoid an influence of noisy sensor data and for a better reproducibility, we set the egocentric obstacle positions manually when testing on the real robot. The measured execution time was averaged over 1000 runs.

The results of the tests on the Nao robot are shown in Tab. I. Overall, the multiresolutional approaches outperform the uniform grid planner clearly. Moreover, there are differences between the local multiresolution and the log-polar grid, as the former has a lower computational complexity. The results show that we should be able to perform path planning in real-time, while executing other required software modules.

In addition to measuring the computing time, we counted the number of node expansions performed by the A*-algorithm. The node expansions for the four test cases are shown in Tab. II. As depicted in Fig. 6, the A*-algorithm expands less nodes in the multiresolutional approaches than in the uniform approach. Even though fewer nodes are expanded in the log-polar grid, the planning time is slightly higher

TABLE I

PLANNING TIME (IN MILLISECONDS) ON THE NAO (UG: UNIFORM GRID, LMG: LOCAL MULTIREOLUTION GRID, LPG: LOG-POLAR GRID).

test case	UG (100%)	LMG	LPG
no obstacles	3.2	0.1 (3%)	0.3 (9%)
ultrasonic	6.3	1.5 (24%)	1.8 (29%)
camera	7.9	1.6 (20%)	3.0 (38%)
ultrasonic & camera	8.0	2.0 (25%)	3.1 (39%)

TABLE II

NUMBER OF EXPANDED NODES OF AN A*-SEARCH (UG: UNIFORM GRID, LMG: LOCAL MULTIREOLUTION GRID, LPG: LOG-POLAR GRID).

test case	UG	LMG	LPG
no obstacles	211	97	75
ultrasonic	1503	241	121
camera	1753	280	153
ultrasonic & camera	1367	239	207

than in the local multiresolution grid. This is caused by the calculation of the logarithm in the coordinate conversion.

B. Path Efficiency

Resource efficiency is only one important aspect of real-time path planning. The main aspect is to find a low-cost path from the start point to a target. To evaluate the quality of the resulting plan, we let the robot walk from random start poses to random targets in simulation. Each start and target pair is used with each of the three planners. The lengths of the resulting trajectories and the required time to execute the plan were measured during this experiment. We evaluated the path efficiency in two test cases, namely without or with static robot obstacles. In the latter case, the field configuration is comparable to a SPL soccer match. The measured lengths are normalized with the length of the corresponding trajectory, achieved by following a path planned with the uniform grid. The path lengths for both test cases are shown in Tab. III.

In the test case without obstacles, the observed difference between the path lengths is not larger than the standard deviation. The second test with obstacles shows that the three different representations lead to similar path lengths. Nevertheless, the log-polar paths are slightly longer and show a higher deviation.

C. Analysis of Real-Robot Paths

In order to test the applicability of our planning approaches in soccer matches, we performed 10 test runs with every grid type on the real robot. The field configuration included four

TABLE III

PATH EFFICIENCY OF THE PATH PLANNING APPROACHES. THE PATH LENGTHS ARE NORMALIZED TO THE LENGTH OF THE WALKING TRAJECTORY WHILE USING THE UNIFORM GRID FOR PLANNING.

	local multiresolution		log-polar	
	mean	sigma	mean	sigma
no obstacles	0.98	0.03	1.03	0.05
with obstacles	0.99	0.04	1.08	0.11

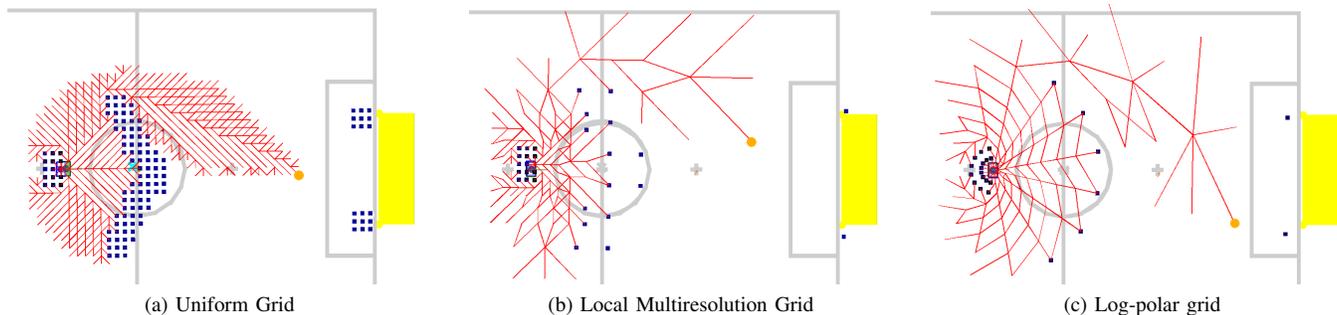


Fig. 6. Node expansions performed by the A*-algorithm in the three grids (red lines). The search expands many more cells in the uniform grid than in the non-uniform grids. Occupied cells are marked with blue dots. The target cell is depicted by an orange dot.

robots positioned close to the center line of the soccer field. We let the robot walk between two target points on opposing sides of the field. The obstacles were perceived using ultrasonic sensors and the cameras. During the tests, we measured the time the robot needed to reach the target points.

As a baseline, the execution time of the straight, obstacle-free path was measured to be 22 s for all representations. The average walking time for both multiresolutional representations was about 37 s. The maximal time measured was 50 s. Repeated re-decisions of the planner due to noisy perceptions caused a motion stall in that case.

The uniform grid path planner, though being sufficiently fast in its average computing time, slowed down the cognition process immensely in more complex situations. This led to collisions and localization errors. Thus, no meaningful data about the paths in the uniform grid could be collected. While employing the path planning in both non-uniform representations, we observed no slow down of the computation.

Entailed by the larger cell size in the further distance of the robot, the resulting paths of the multiresolutional planners were mostly avoiding the robot obstacles well in advance. Only in cases where robots were perceived late, paths through narrow gaps between the robots were planned.

VI. CONCLUSION AND FUTURE WORK

In this paper, we evaluated two approaches to path planning which are applicable to soccer robots with relatively low computational power. Both approaches make use of properties found in the soccer domain. An important property is the lack of static obstacles in the environment of the soccer field. For this reason, it allows planning at a coarse resolution for regions which are far from the robot. Furthermore, one can expect to find a valid plan refinement in order to avoid dynamic obstacles while approaching them. Consequently, our grid representations employ a decreasing resolution for distant parts of the environment.

As virtually all obstacles are dynamic, it is likely that the situation in distant cells will have changed at the time a plan refinement will be necessary. Therefore, we are convinced that approximate planning with continuous and fast replanning

is superior to slower exact planning. Our experiments show that while multiresolution approaches provide paths of similar quality to full-resolution planning, the computational efficiency increases immensely. Moreover, with the non-uniform grid representations, the path planning can be performed two to four times faster than with a uniform grid, in average. The real-robot experiments reveal that this speedup facilitates real-time planning on the Nao. On our website¹, we provide modules of our planning algorithms compatible with the framework released by the B-Human team after RoboCup 2010.

In future work, we aim at integrating more robust robot perception. Furthermore, the estimation of possible future obstacle positions is likely to improve plans with regard to the need of necessary replanning.

ACKNOWLEDGMENT

This work was partially funded by the German Research Foundation (DFG), grant BE 2556/2-3.

REFERENCES

- [1] S. Behnke and J. Stückler, "Hierarchical Reactive Control for Humanoid Soccer Robots," *International Journal of Humanoid Robots (IJHR)*, vol. 5, no. 3, pp. 375–396, 2008.
- [2] L. Kaelbling and T. Lozano-Pérez, "Hierarchical Task and Motion Planning in the Now," *MIT-CSAIL-TR-2010-026*, 2010.
- [3] S. Behnke, "Local multiresolution path planning," *RoboCup 2003: Robot Soccer World Cup VII*, pp. 332–343, 2004.
- [4] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. MIT Press, 2001.
- [5] P. Sermanet, R. Hadsell, M. Scoffier, U. Muller, and Y. LeCun, "Mapping and planning under uncertainty in mobile robots with long-range perception," in *Proc. Intelligent Robots and Systems (IROS'08)*, 2008.
- [6] L. Longega, S. Panzieri, F. Pascucci, and G. Ulivi, "Indoor robot navigation using log-polar local maps," in *Prep. of 7th Int. IFAC Symp. on Robot Control*, 2003, pp. 229–234.
- [7] RoboCup Technical Committee, *RoboCup Standard Platform League (Nao) Rule Book*, 2010.
- [8] Aldebaran Robotics, *Nao User's Guide Ver 1.6.0*.
- [9] —, *Nao Academics datasheet*.
- [10] T. Röfer, T. Laue, J. Müller, O. Bösch, A. Burchardt, E. Damrose, K. Gillmann, C. Graf, T. J. de Haas, A. Härtl, A. Rieskamp, A. Schreck, I. Sieverdingbeck, and J.-H. Worch, "B-human team report and code release 2009," 2009, only available online: http://www.b-human.de/file_download/26/bhuman09_coderelase.pdf.
- [11] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice hall, 2009.

¹www.ais.uni-bonn.de/download/splplanning