

Local Multiresolution Path Planning in Soccer Games Based on Projected Intentions

Matthias Nieuwenhuisen, Ricarda Steffens, and Sven Behnke

Autonomous Intelligent Systems Group, Institute for Computer Science VI
University of Bonn, Germany

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 local-multiresolution 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 extend the multiresolution approach to the time dimension and we predict the opponents movement by projecting the planning robot's intentions.

1 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.

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 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.

In arbitrary situations, it is hard to predict the movements of dynamic obstacles without tracking them. Thus, these movements are often modelled as random noise. In contrast, the movements of the opponent in soccer games are aimed at scoring goals. We employ this fact to gain a better estimation of probable obstacle trajectories by projecting the intentions of the planning robot to other field players.

2 Related Work

Our humanoid soccer team uses a hierarchical reactive approach to control the robot's motion [2]. 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 [4]. 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.

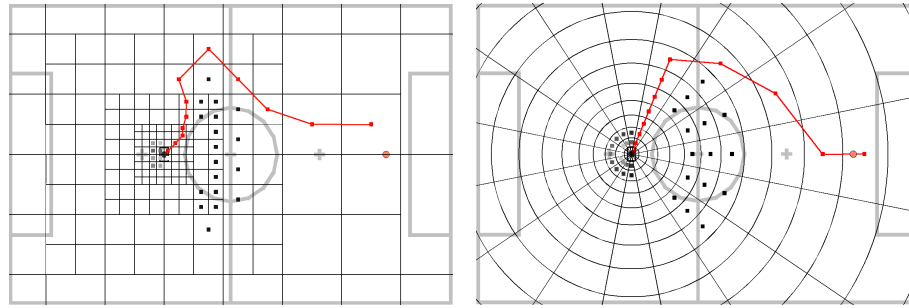
A method for resource-saving path planning is the local multiresolution Cartesian grid [1]. It employs multiple robot-centered grids with different resolutions, and nests them hierarchically while connecting them through adjacencies. This representation resembles our approach to approximate plan steps 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 [11], polar coordinate based grids can be found for egocentric robot motion planning in the literature [3], [5]. 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 [10].

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.

3 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 de-



(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. 1. 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.

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 approximately 14 Hz . The ultrasonic sensors are located at the robot's chest covering an angle of approximately 110° in front of the robot. The detection range is from 300 mm to 700 mm ¹.

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 memory². 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.

Our software is based on the framework of the German SPL-team B-Human [8]. 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.

4 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,

¹ Nao User's Guide Ver 1.6.0, Aldebaran Robotics

² Nao Academics Datasheet, Aldebaran Robotics

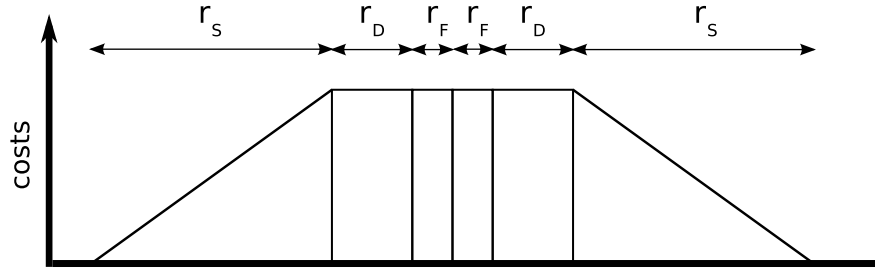


Fig. 2. Obstacles are modeled as a fixed radius r_F and a distance dependant radius r_D with maximum costs and a safety margin r_S with decreasing costs.

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.

4.1 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. Perceived obstacles are initialized with a radius and costs. The radius consists of a static part representing the obstacle radius plus a component that increases linearly with the distance to the grid center. This component represents the uncertainty in distant measurements and the obstacle movements. To keep the overall costs of an obstacle constant, the costs are reduced accordingly. 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. 2. The positions of visually perceived obstacles and sonar readings are maintained in different representations outside of the grid and incorporated into it during planning.

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. [9]).

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.

4.2 Local Multiresolution Grid

An efficient path planning algorithm is local multiresolution path planning [1]. 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 $[\frac{M}{4} : \frac{3M}{4}] \times [\frac{M}{4} : \frac{3M}{4}]$ 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. 1a 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 coarser grid have seven neighbors at the edges and six neighbors at the corner. Likewise, the grid cells at the border to the finer 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.

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.

4.3 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 is 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. 1b.

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.

4.4 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. 3). 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 the weighted vector of position and angle of the waypoint relative to the robot.

5 Evaluation of the Representations

5.1 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. The used sensors influence the planning time, because sonar sensors can only perceive close obstacles and have a low

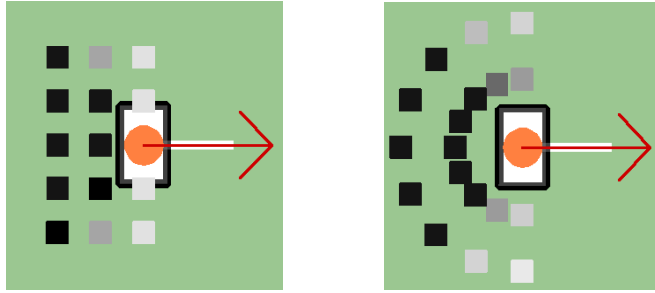


Fig. 3. An artificial obstacle leading to preference of paths towards the robot’s current walking direction to avoid sudden directional changes. Cells with higher costs are darker. The red arrow is the gait target vector (left: Cartesian grid, right: polar grid).

Table 1. Planning time (in milliseconds) on the Nao (UG: uniform grid, LMG: local multiresolution grid, LPG: log-polar grid).

test case	UG	LMG	LPG
no obstacles	2.9	0.4	0.3
ultrasonic	12.2	0.8	1.2
camera	18	2.3	3.0
ultrasonic & camera	23	3.0	3.5

angular resolution. This results in only few, but large obstacles in the vicinity of the robot. In every test case, the target was 4000 mm in front of the robot and five other robots were on the field. 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 and used ground truth data in simulation. The measured execution time was averaged over 1000 planner runs.

The results of the tests on the Nao robot are shown in Tab. 1. 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.

6 Dynamic Planning with Intention Projection

Most of the obstacles apparent on the soccer field are not static. As described before, we take this into account by smoothing distant obstacles. If the movement of the other robot’s can be estimated in advance, it is possible to avoid future positions of these. Naturally, it is not possible to reason about the exact behaviors and targets of the opponent team. Thus, we have to apply situation specific assumptions for probable movements of the opponent. For example, if our robot intends to reach a position where it can get ball possession, this intent is projected to the opponent. Thus, we assume it will move to a position where it can control the ball, too. This assumption is reasonable, as opponents with the same intent as the planning player will more likely intersect the planned path than robots with different intents.

Our dynamic obstacle model extends the static obstacle model by introducing an assumed target relative to the obstacle and a velocity vector v . To be consistent with the multiresolutional representation of space, the obstacle radius is increased and the costs are decreased with increasing distance to the robot. We assume that the opponent may either keep standing still or move to the target with any speed up to a maximum v_{max} . This results in a uniform probability distribution of the opponent's position along a vector $v_{max} \times t$. To represent this the representation of the obstacle grows into the direction of its movement up to the assumed target with advancing time t . The overall costs of the obstacle are kept constant. Hence, the costs of the obstacle decrease.

To represent the resulting different shapes of obstacles at different times, we extend the grid representation to the time dimension. The resulting three-dimensional grid is discrete in time and this discretization may not necessarily equal the space discretization for arbitrary robot speeds. To take this into account, we explicitly calculate the average time the robot needs to follow the planned path up to the current grid cell and discretize it afterwards to determine whether edges connecting nodes within the same time level can be followed.

Figure 4a shows an example where the planned paths with and without dynamic planning are qualitatively different. Our robot aims to reach a position next to the ball to dribble it towards the opponent goal. Therefore it has to walk around the ball. Another player is approaching the ball from the left bottom side of the map. Utilizing a planning algorithm not taking the directed movement of this robot into account, the robot would try to pass the ball at the side the opponent is coming from, possibly having to avoid it later on. The dynamic approach plans the same shortest path, if the opponent is still far away. Otherwise, the ball is passed on the other side, where an intersection with the opponents path is unlikely.

Obviously, adding the time dimension makes the planning computationally more involved. For the uniform time discretization we use a 16 s lookahead, discretized into 1 s steps. This results in a local-multiresolution grid with 4096 cells. Thus, we reduce the complexity, following the same ideas of a multiresolutional representation as in the space dimension. The minimal time an opponent robot needs to reach a cell corresponds to the distance of that robot to the cell. Here, we have three qualitatively different cases to consider:

- The robots are close to each other and to the cell. Thus, the cell is represented at a fine spatial resolution and an intersection of the robot's paths may occur soon. A finer temporal resolution is necessary.
- The cell is further away. Hence, it is represented at a coarse spatial resolution and coarse temporal resolution is sufficient, regardless of the position of the opponent.
- The cell is close to the planning robot, but the opponent is further away. In this case a coarse resolution is sufficient to determine, if an intersection of paths in this cell is likely.

Figure 4b depicts snapshots of the ball approaching example using a multiresolutional time representation. The discrete time steps t_m are defined as

$$t_m(i) = \begin{cases} i = 0, & 0 \\ i > 0, & 2^i \end{cases}$$

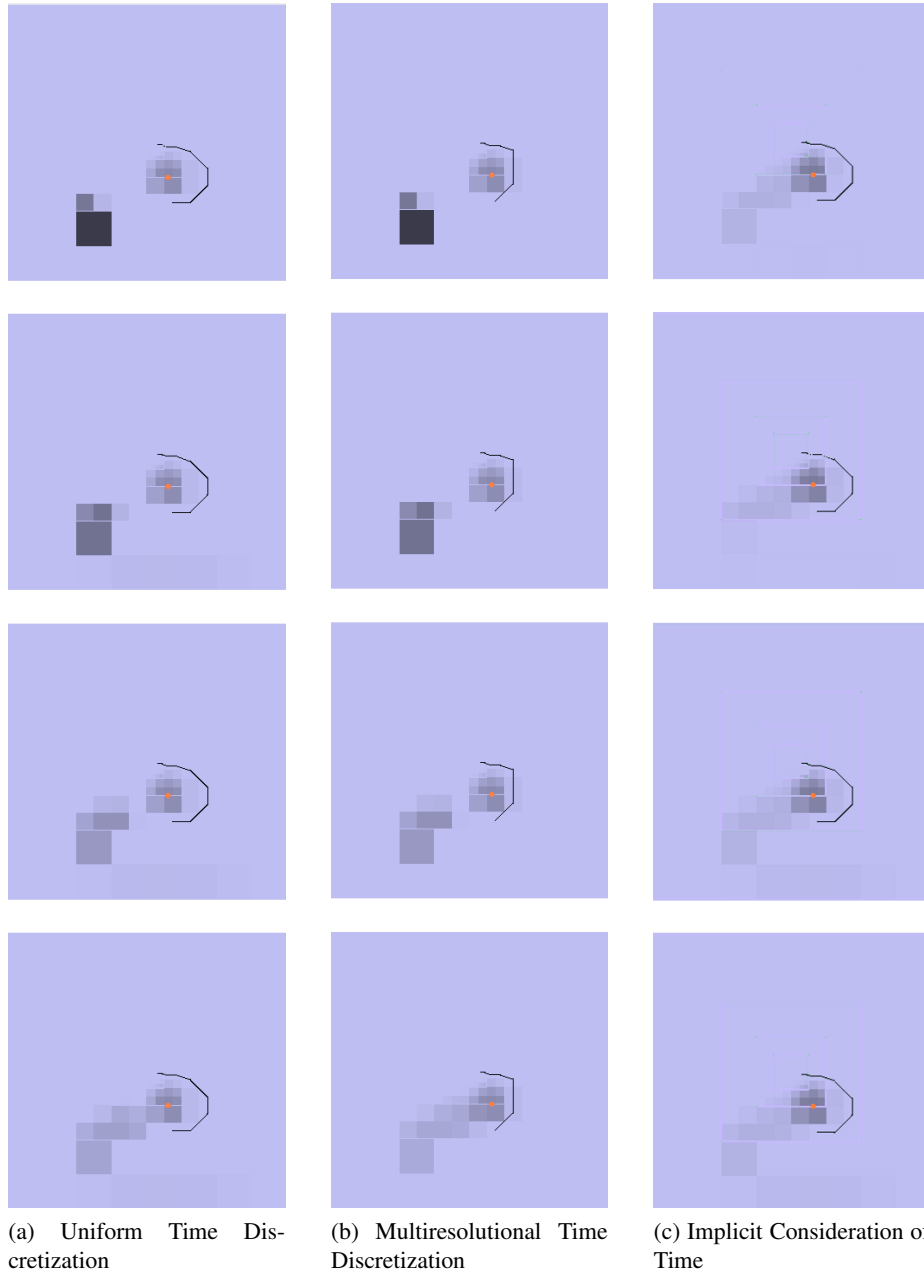


Fig. 4. The planning robot aims to reach a position behind the ball to dribble it to the goal. An opponent is detected and its position is propagated over the time by elongating the corresponding obstacle shape. At the beginning costs of cell nodes are calculated using a circular obstacle model. With increasing time, the obstacle representation becomes more elongated until it reaches the target, i. e. a position behind the ball opposing our target. The time increases from the top ($t=0$) to the bottom images ($t=8$ s). The left image series shows the uniform time discretization, the middle series the multiresolutional time discretization and the right series the implicit consideration of the time.

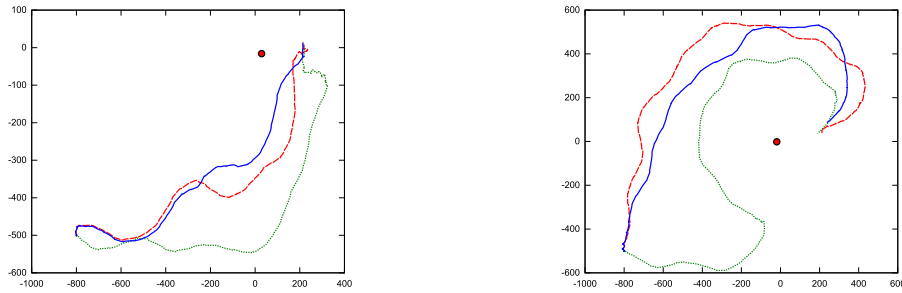


Fig. 5. Comparison of the robot’s trajectories while using the planner without time consideration (green/dotted), with uniform time discretization (blue/dashed), and with implicit time incorporation (red/solid). The target is a point behind the ball (circle). An opponent is approaching from the bottom right side. In the example on the left side the opponent is further away than in the example depicted on the right side. Both planners that propagate the opponent’s position plan qualitatively similar path. While following the plan without representation of the time, the robot had to replan to avoid a collision with the opponent in the case depicted in the right figure.

The resulting grid contains 1280 cells and covers the same time and space as the uniform grid.

As the speed of the planning robot is bound, the time necessary to reach cells increase linearly with their distance to the robot, defining a lower envelope of the reachable cells. Thus, major parts of the three dimensional grid cannot be reached. Especially, the fine grained temporal resolution steps are mostly below this envelope. For the planner it is of primary importance if a path through a cell may possibly cause a collision in the future. Shortest paths often start with long straight segments towards the target. In the case of an apparent obstacle, the segment is split into two segments with an intermediate point that modifies the path to be collision free. The time to follow the first segment can be approximated by a linear function of the distance between robot and endpoint of that segment. This leads us to an implicit incorporation of the time dimension into our obstacle model. We determine the minimal distance to the line segment between the opponent and its assumed target. With the linear time approximation function, we get the approximate time of the probable intersection of the two paths. This approximation is used to estimate the distribution over the possible positions of the obstacle on the field. As time is only implicitly taken into account for planning, the planning problem stays two-dimensional.

We compared the planning time on the Nao robot for the ball approaching example. To plan a path around the ball without considering the opponents movement took on average 1.6 ms , with 16 uniform time discretization steps 55.4 ms . The multiresolutional discretization with 5 steps took 7.9 ms and the implicit approach took 3.3 ms . With every representation of the time component it was possible to use the shortest path, if the opponent was sufficiently far away and to avoid the opponents path if it was closer (see Fig. 5).

7 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. Furthermore, the estimation of possible future obstacle positions is likely to improve plans with regard to the need of necessary replanning. Thus, we use assumptions about the behavior of soccer playing robots to estimate their trajectories.

The real-robot experiments reveal that the speedup facilitates real-time planning on the Nao.

Acknowledgment

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

References

1. Behnke, S.: Local multiresolution path planning. Robocup 2003: Robot Soccer World Cup VII pp. 332–343 (2004)
2. Behnke, S., Stückler, J.: Hierarchical Reactive Control for Humanoid Soccer Robots. *International Journal of Humanoid Robots (IJHR)* 5(3), 375–396 (2008)
3. Borenstein, J., Koren, Y.: The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots. *Robotics and Automation, IEEE Trans. on* 7(3), 278–288 (1991)
4. Kaelbling, L., Lozano-Pérez, T.: Hierarchical Task and Motion Planning in the Now. MIT-CSAIL-TR-2010-026 (2010)
5. Lagoudakis, M., Maida, A.: Neural maps for mobile robot navigation. In: *IJCNN '99*
6. Longega, L., Panzieri, S., Pascucci, F., Ulivi, G.: Indoor robot navigation using log-polar local maps. In: *Prep. of 7th Int. IFAC Symp. on Robot Control*. pp. 229–234 (2003)
7. RoboCup Technical Committee: *RoboCup Standard Platform League Rule Book* (2010)
8. Röfer, T., Laue, T., Müller, J., Burchardt, A., Damrose, E., Fabisch, A., Feldpausch, F., Gillmann, K., Graf, C., de Haas, T.J., Härtl, A., Honsel, D., Kastner, P., Kastner, T., Markowsky, B., Mester, M., Peter, J., Riemann, O.J.L., Ring, M., Sauerland, W., Schreck, A., Sieverdingbeck, I., Wenk, F., Worch, J.H.: *B-Human Team Report and Code Release 2010*, www.b-human.de/file_download/33/bhuman10_coderelease.pdf
9. Russell, S., Norvig, P.: *Artificial Intelligence: A Modern Approach*. Prentice hall (2009)
10. Sermanet, P., Hadsell, R., Scoffier, M., Muller, U., LeCun, Y.: Mapping and planning under uncertainty in mobile robots with long-range perception. In: *Proc. of IROS* (2008)
11. Thrun, S., Burgard, W., Fox, D.: *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. MIT Press (2001)