Local Goal Model for a Humanoid Soccer Robot
Extended Abstract

Heinrich Mellmann and Marcus Scheunemann *

Institut für Informatik
LFG Künstliche Intelligenz
Humboldt-Universität zu Berlin, Germany
{mellmann,scheunem}@informatik.hu-berlin.de

1 Introduction

Many tasks of a mobile robot, e.g., navigation, require the knowledge of the positions of the objects in the surrounding environment. This is especially challenging for the robots which perception is based on a directed visual system, e.g., a camera with a limited view angle. The incomplete and noisy sensor information leads to the uncertainty in the robots' belief of the world. An appropriate model of the world is necessary to enable the robot to make plans and to realize complex behavior.

In general there are two possible approaches to model the world: the objects relative to the robot (local coordinates) or relative to the environment (global coordinates). Depending on the task, the first or the former one may be more suitable. Basically, the objects can be divided in two categories: static and dynamic ones, e.g., a goal post and a ball. In the most cases, it is more suitable to consider the dynamic objects relative to the robot itself, whereas for the navigational tasks the position of the robot itself is modeled relatively to a certain global coordinate system.

As described in the section 2, the context of *RoboCup* provides a very good environment to examine the different modeling approaches. Here, the static structure of the environment, i.e., positions of the lines , goals etc., is known. Thus, the common approach is to model the position of the robot relatively to the field. The positions of the particular objects can be retrieved using the knowledge of the fields' geometry. This approach provides a global solution for the modeling of the static objects in this context. However, this way may also has striking drawbacks independently of the particular algorithms used. A lot of tasks can be solved using only some information of a particular object. Good examples are provided by problems involving relations between particular objects, e.g., deciding whether the ball is outside of the middle circle. If the robot recognizes the ball and the middle circle, he can make a decision without knowing his precise position on the field. Of course we need a model of the middle circle in this case,

* Both authors contributed equally to this work. The authors would like to thank the members of the *Nao Team Humboldt*.

CONCURRENCY, SPECIFICATION AND PROGRAMMING
M. Szczuka et al. (eds.): Proceedings of the international workshop CS&P 2011
September 28-30, Pułtusk, Poland, pp. 353-360

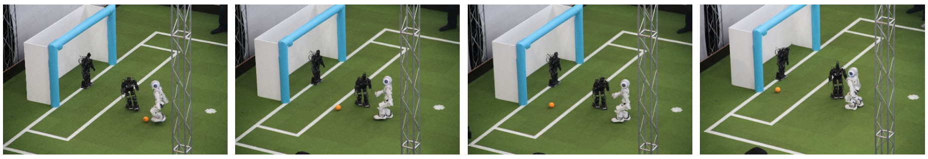

Fig. 1. This sequence of states describes a typical necessity of local knowledge. The robot tries to kick the ball inside the goal with too less force and the ball remains in front of the goal. Now, when the robot approaches the ball, there is no need for knowing its exact position on the field. The robot just has to know weather the goal post close to the ball is the goal's right or left one. Additional information, e.g., line pieces or former behavior, leads to a sufficient model of the world.

because of the sensory noise and because the robot may not see the circle permanently. However, it can be done much more accurate in this particular case. Especially, when the ball and the circle are seen at the same time, the errors are correlated and the relations between them are preserved. Additionally, the robot does not need to search for other information to determine its full position, so it may save computations and time until the task can be solved. Obviously, in this example it make sense to model the circle relatively to the robot, i.e., the same as the ball. Generally, we can say that local models may be more suitable to model static objects in certain situations.

The most currently popular approaches are based on the probabilistic Bayes-filtering. In particular, the most solutions within the RoboCup domain are using these idea. Here it is tried to solve the global localization problem by integrating the sensory data into a probability distribution describing the position of the robot on the field. Some examples for successful implementations are given in [1, 5, 6]. So far there have been only few research done on local modeling cf. [3].

In this paper we investigate how static objects can be modeled locally on example of a *goal* in the *RoboCup* context. In particular we present our algorithms based on the idea of the *particle filters* [6–8] for the modeling of these objects in the section 3. After the illustration of the first experimental results in the section 4 we also discuss how these models may be integrated in a complete (local) behavior. At the end we conclude the results and outline the direction for the future work.

2 Environment and Platform

In this section we briefly discuss the technical setup of the experimental environment. In particular, the RoboCup, the experimental humanoid robotic platform Nao as well as his perceptual capabilities are presented.

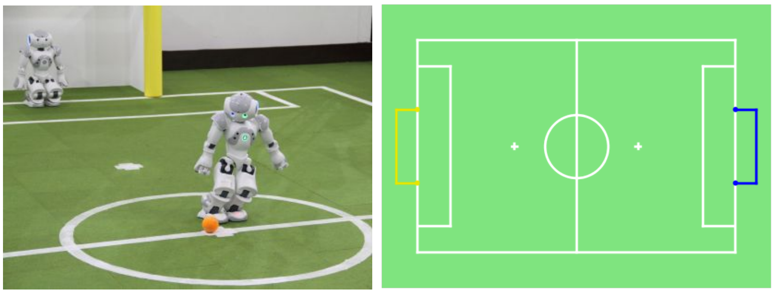

Fig. 2. Robot *Nao* playing soccer in the *Standard Platform League* of the *RoboCup*. Here, the robots have to play soccer in teams autonomously. The field is color coded to make image processing easier.

2.1 RoboCup

The *Robot World Cup Initiative* (RoboCup for short) was founded in 1997 and is an international robotics competition. The aim is to develop autonomous robots with the intention of promoting research in robotics and artificial intelligence.

The work discussed in this paper is tested in the *Standard Platform League (SPL)*, where our team *Nao Team Humboldt* takes part in. The environment of the field corresponds to fixed geometrical rules. There are white lines situated similar to a real soccer field and two different colored goals on a green field, as you can see in Figure 2. Furthermore, within the *SPL* all teams are using the same robot platform *Nao*, which will be described in the next section. Because of this given world, we can focus on investigate modeling approaches in a dynamic world with known objects, instead of dealing with engineering issues.

2.2 Nao Robot

The humanoid robot *Nao* has a very articulate body, it is 58cm of size and weighs about 4.8kg including the battery. Each robot has 21 degrees of freedom. The joints are actuated by DC motors and the platform is equipped with a low power and low consumption *Geode LX 800* processor with just 500 MHz. Since the CPU processing power is quite limited compared to the robot's physical structure, it is challenging to implement very complex algorithms for modeling. Therefore simplicity in design is the preferred approach due to a lack of computational power. The robot's head is equipped with two vertical aligned VGA cameras. It is only possible to operate one of them at a time. The maximum refresh rate of the cameras is 30 fps. Further the robot is equipped with 3-Axes accelerometer, a 2-Axes gyro and four force resistive sensors (FSR) in each foot. This data can be used to estimate the orientation of the robot's body relatively to the ground. More detailed information can be found in [2].

2.3 Visual Perception

The main sensor of the robot is the camera in his head. Based on the colors the image processing is able to detect all important objects in the images, e.g., goal posts, lines, ball, other players. Using the knowledge of the robots kinematics and the proprioceptive sensors accelerometer, gyro and FSR it is possible to estimate the position of these objects relative to the local coordinates of the robot. These relative positions are used as input for the modeling algorithms described in the next sections. In general we can consider these percepts as measurements of certain abstract sensors, e.g., a ball sensor providing the distance to the ball. Of course, those sensors are affected by some uncertainty due to noise in the image, the discretization and the errors in the image processing. It should be underlined that this kind of uncertainty may be much more complicated compared to a simple sensor like gyro. In particular, we may have discontinuities, e.g., a false detection of the ball. This makes it difficult to model these sensors using classical filtering approaches.

3 Local Goal Model

The first question concerns the representation of the goal. If we assume the width of the goal fixed, three parameters are enough to describe its pose, i.e., position of a point (e.g., left post) and orientation. However, one of the main difficulties we have to deal with is that in the most cases the robot can see only one of the posts, i.e., the whole state of the goal cannot be observed. Thus, we decided to model each of the goal posts as a separate object. The model of the whole goal is estimated based on these models.

To model the single post a particle filter is used, i.e., we need several filters to maintain several posts simultaneously. The basics and the according theoretical background can be fond in [7]. Since the posts are ambiguous we have to resolve the assignment problem for the percepts. Due to possible false detections we also have to assume that there may be more than two different posts detected.

A goal post percept can be described in polar coordinates as a vector $(\alpha, d) \in \mathcal{X}_p := [-\pi, \pi] \times \mathbb{R}_+$ in the local coordinates of the robot, where d is the distance to the goal post and α the horizontal bearing angle. The internal model (hypothesis) of a single goal post is represented by a final set of particles $H \subset \mathcal{X}_p$. There are fixed lower and upper bounds for the number of particles in a particle set $m, M \in \mathbb{N}$ respectively, i.e., $m \leq |H| \leq M$. The whole model consists of a Set of post hypotheses $\mathcal{H} := \{H_1, ..., H_n\}$ and a percept buffer $B \subset \mathcal{X}_p$.

Algorithm

The Algorithm is triggered by a new image. Beside the post percepts $P := \{p_1, ..., p_k\} \subset \mathcal{X}_p$ the robot gets odometry information o, i.e., a 2D coordinate transformation describing his motion since the last update. In each cycle we execute the following steps.

1. Update all particle filters in $\mathcal{H}$ and the buffer B by odometry information, i.e., apply the transformation o to each element of the sets;
2. For all incoming percepts $p \in P$: classify whether p fits to an existing filter in $\mathcal{H}$ by a specified threshold, in case there is such $H \in \mathcal{H}$ update H by sensory data p and remove p from P, i.e., $P := P \setminus \{p\}$;
3. Check for all existing filters $H \in \mathcal{H}$ if their *weight* exceed a defined threshold, otherwise discard this hypothesis. Thereby, the weight of a filter is calculated by a weighted integral of the prior weightings;
4. Resample particles of existing filters $H \in \mathcal{H}$;
5. Put all remaining percepts in a buffer $B := B \cup P$;
6. Remove old percepts from the buffer$B$;
7. Cluster the buffer by euclidean distance and check if it contains a *large* cluster $C \subset B$ with $|B| > m$. Use this cluster to initialize a new filter H and add it $\mathcal{H} := \mathcal{H} \cup H$;

It should be remarked, that in the step 2 only one filter is updated by a percept, so we explicitly solve the assignment problem. To estimate the model of the whole goal we select two filters $H_1, H_2 \in \mathcal{H}$ which have the best weightings and satisfy the distance between the two goal posts. Obviously, the whole goal can only be estimated when such filters are available. In this case we assume the means of H_1 and H_2 to be centers of the goal posts.

This is of course a loose description of the algorithm. In the detail there are still some issues to be solved. In particular, it is not clear which measure is suitable in the step 2. Another open issue concerns the aging criteria for the filters in the steps 3 and 6. Finally, calculating of the whole goal (to be more general: incorporation of relations) is also a subject for the current research.

4 Experiments

In order to test the performance of our modeling approach we performed some experiments on the rial robot. In the first experiment we investigate the quality of the goal percepts provided. The second experiment aims to test the qualitative ability to keep track of the goal model in a simple game situation.

4.1 Goal Post Perception

In this experiment we examine the error of the goal percepts and its distribution. This test where made beforehand to discuss a suitable model approach. The right image of Figure 3 shows the experimental setup. The robot is situated in front of the right goal post with a fixed distance of 3m. While recording the percepts, the robot permanently stepping on the same position. This moving and the distance causes the left image of Figure 3. It shows the distribution of goal post percepts. As you can see, the measured goal posts' angle is not that erroneous as the measured position. This is because small errors in the tilt angle of the body may result in large projection distance errors. There was already an investigation about the bearing based distance estimation error for fixed camera height in [4].

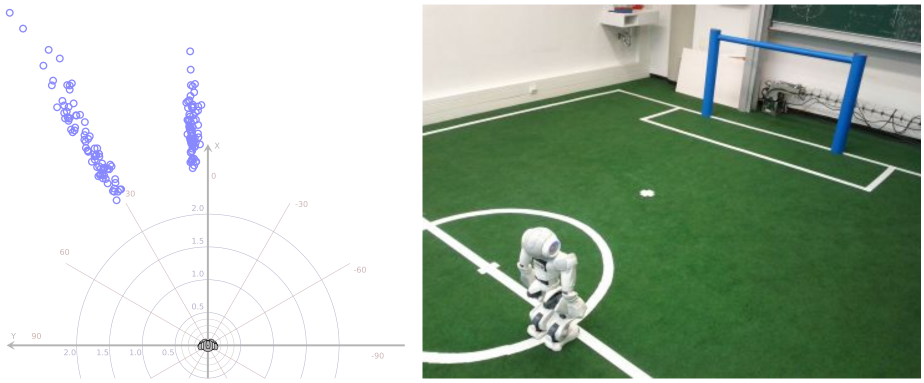

Fig. 3. Goal posts perceived by the robot which is stepping on a spot and turning his head around during a time of 30s. Thereby, the robot is situated in front of the right post with the distance of 3m as shown in the right image. The left figure illustrates the percepts of the seen goal posts in the local coordinates of the robot.

4.2 Approaching Goal

The aim of this experiment is to test the qualitative modeling performance in a simple dynamic game situation. The general setup of the experiment is similar to the situation shown in Figure 1, i.e., the robot approaches the ball which is located nearby a goal post.

In the initial situation of the experiment the robot is located in a certain distance from the goal, while the ball is placed nearby a goal post. From this position the robot is able to observe the whole goal while looking at a ball. During the experiment the robot's task is to follow the ball and to keep track of the goal model. Thereby the robot is always looking at the ball. During the experiment the ball is moved slightly around to simulate some game dynamics.

The Figure 4 visualizes the results of an integrated experiment. The upper and the lower rows illustrate the initial and the final state of the experiment from the view of the robot respectively. On the left side the view of the robot's camera is shown. On the right side you can see the robot's internal belief of the goal position.

As you can see, at the beginning (upper row in Figure 4) the robot already holds two hypotheses about two different goal posts represented by two differently colored heaps of particles. The black line between both filters illustrates the robot's inner goal belief. After approaching the ball the robot still hold its hypotheses, although, he was not able to both goal posts permanently (cf. Figure 4, lower row).

5 Conclusion and Future Work

In this paper we presented an algorithm for local modeling of a goal within the RoboCup domain. Our first experiments provide a proof of the concept, i.e., the

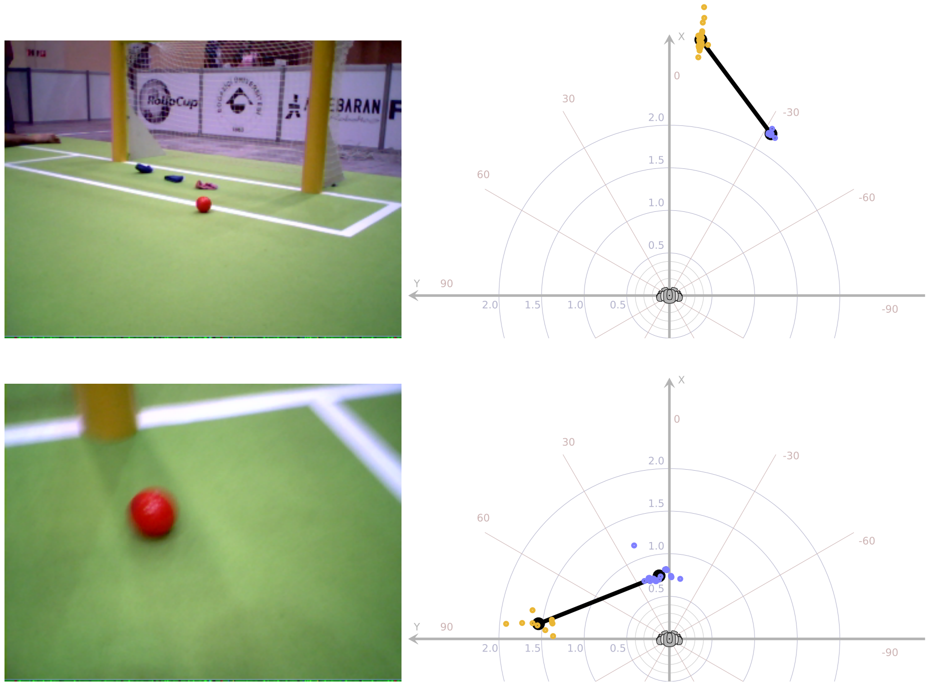

Fig. 4. Goal modeling during approaching the ball. Left column: scene from the view of the robot. Right column: the local model of the goal. Different clusters of particles are marked with different color. The black bold bar visualizes the estimated pose of the goal.

robot is able to localize the goal and track it over the time. This algorithm has been successfully used at the recent RoboCup competitions on a simulated and on a real robot.

Although the presented modeling algorithm is not really new, the idea of local modeling didn't gain much attention so far. Based on the experience we gained from these experiments we believe that this way of modeling may provide a more rich, stable and flexible representation of the environment. In particular, it will provide more detailed and reliable information for interaction with the world.

As already mentioned, the presented work is in progress. So, to make more reliable and precise conclusions there is still a lot of work to be done. In particular, development of appropriate numerical and analytical quality measures is the next step to be done. To take a outlook into the near future, we are currently working on further partial object models, e.g., lines. The next step will be the integration into a common global model. Thereby, the major question is handling of relations between singe objects, e.g., distance between the two posts of the same goal.

References

1. D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and Eleventh Conference on Innovative Applications of Artificial Intelligence (AAAI)*, pages 343–349. The AAAI Press/The MIT Press, 1999.
2. D. Gouaillier, V. Hugel, P. Blazevic, C. Kilner, J. Monceaux, P. Lafourcade, B. Marnier, J. Serre, and B. Maisonnier. Mechatronic design of nao humanoid. In *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pages 769 –774, May 2009.
3. P. Guerrero and J. R. del Solar. Improving robot self-localization using landmarks' poses tracking and odometry error estimation. In U. Visser, F. Ribeiro, T. Ohashi, and F. Dellaert, editors, *RoboCup 2007: Robot Soccer World Cup XI*, volume 5001 of *Lecture Notes in Computer Science*, pages 148–158. Springer, 2007.
4. H. Mellmann, M. Jüngel, and M. Spranger. Using reference objects to improve vision-based bearing measurements. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems IROS 2008*, pages 3939–3945. IEEE, 2008.
5. M. J. Quinlan and R. H. Middleton. Multiple model kalman filters: A localization technique for robocup soccer. In J. Baltes, M. G. Lagoudakis, T. Naruse, and S. S. Ghidary, editors, *RoboCup*, volume 5949 of *Lecture Notes in Computer Science*, pages 276–287. Springer, 2009.
6. T. Röfer and M. Jüngel. Vision-based fast and reactive monte-carlo localization. In D. Polani, A. Bonarini, B. Browning, and K. Yoshida, editors, *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA)*, pages 856–861. IEEE, 2003.
7. S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21:93–109, 2000.
8. S. Thrun. Particle filters in robotics. In *in Proc. 18 th Conf. Uncertainty in Artificial Intelligence*, pages 1–4, 2002.