Object Modeling in Dynamic Environments by Mobile Agents

Daniel Göhring

Institut für Informatik, LFG Künstliche Intelligenz,
Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany
http://www.aiboteamhumboldt.com

Abstract. Modeling of the environment based on sensor data plays an important role for research works in artificial intelligence, especially for robotics. This modeling task can be split into two subtasks for the robots - self localization and object tracking. In self localization the robot uses landmarks to determine its own position at a certain area. Object modeling and tracking describes the process of observing and processing object information, making it possible to predict future object states. One problem for humans as for robots is that at every time only a small part of the complex environment can be observed and that sensor data are not very accurate. Furthermore object modeling is not just only a passive process, observing the surrounding as in many approaches, but must take into account the interactions of the agent with its environment as well.

This paper presents a fast and robust approach for object modeling in the RoboCup [2] domain, based on Rao-Blackwellized particle filters [13]. Rao-Blackwellized particle filters combine the positive aspects of analytical filter algorithms e.g. Kalman filters (accuracy, calculation speed, optimality for linear processes) [5][10][18] with multi modal, nonlinear filtering capabilities of Monte Carlo filter algorithms.

In this paper the Kalman filter was used to model one part of the environment state - the linear ball motions. The nonlinear ball motions like kicking, deflecting on objects or bouncing of the border were modeled with particles. The object modeling approach was extended to the use of negative information, making it possible for the robot not only to track objects but also to find objects back quicker than by arbitrary search. Thereby the usage of obstacle information allowed to distinguish between visible and invisible areas.

1 Introduction

Modeling of the environment plays an important role in mobile robotics as in many other sciences. Moreover it is essential for long time planning. Without a detailed modeling of the environment states it is hard to predict future states. RoboCup with its highly dynamic state changes provides a very special and useful test bed for the development of new modeling algorithms because the applied algorithms have to be fast, robust against noisy input data, and efficient.

1.1 Related Work

During the last years, research for localization and modeling tasks was increased. Probabilistic methods turned out to be very promising for this purpose. Already in 1998 an active sensing architecture, based on a Markov model was implemented to let a robot localize on an office floor, Fox [7]. Other works describe the usage of particle filters as a new version of Markov localization for robot localization [8][3]. E.g. Fox describes the advantages of **M**onte-**C**arlo **P**article **F**ilters (denoted: MCPF) against former analytical methods [4]. Based on MCPF, in [17] an application is described for particles due to robot localization in RoboCup. Because the number of particles should always be as small as possible, Fox introduced a method in [6] how the number of particles can be dynamically adapted to the current situation with the help of KLD-Sampling[1]. Recent work copes with the improvement of Monte-Carlo algorithms. A new generation of particle filters, the **R**ao-**B**lackwellized **P**article **F**ilters (denoted RBPF) is an important subject of current research. Thus Khan describes in [12], how the dimension of the state space can be reduced by dividing the state space into subspaces, which are modeled analytically. Andrieu and Doucet showed in their work [1] the advantages of RBPF compared to other methods like MCPF.

2 Motivation

Ball modeling in the legged league of the RoboCup is a challenging task and thus provides a useful basis for several research projects in the object modeling and tracking domain. For a robot it is important to know, where it is localized on the field and on which places objects are, which is necessary for interaction with the environment.

During a RoboCup game, ball position, velocity and the own position of the robot are influenced by permanent changes. Especially the position and velocity of the ball tend to behave highly non-linear which makes the use of Kalman filters tricky. Examples for the non-linear ball motion are given whenever the ball is kicked by a robot, when a ball bounces off another robot or the (former) field border. In contrast to that the speed and position of a free moving ball behave usually linear. A common approach for creating the world model for the robot[2] is usually done by separating the modeling task into two subtasks: firstly modeling the robot's position and secondly modeling objects as the ball [15]. Uncertainties of the self localization are not considered for the object tracking task. But theory and experience show, that the high interaction of the ball with its environment demand a combined model for the self localization and the tracking task.

The presented approach is based on RBPF which combines the modeling of the robot position as well as the ball position. The linear motion parts of the ball are modeled by Kalman filters because Kalman filters are optimal filters for linear

[1] KLD stands for error estimation by **K**ullback-**L**eibler **d**istance.

[2] a world model contains different objects as the own position, positions of objects from the environment

unimodal motions with Gaussian uncertainty. Robot position and interactions of the ball with its environment, resulting in non linear motions are modeled by particle filters, which are useful for multi modal and noisy object modeling. Thereby the advantages of analytical approaches as (fast and accurate) can be combined with particle filters which tend to be useful for unknown distributions and robust against noisy input data.

3 Rao-Blackwellized Particle Filter

3.1 Kalman Filter

Kalman filtering is performed in two steps:

1. Prediction of the next state from the current state and
2. Update of the predicted data with the help of sensor data and the Kalman-gains.

If data about executed motions u_{t-1} is available, the new a priori state $Bel^{-}(x_t)$ is predicted as follows:

$$Bel^{-}(x_t) \longleftarrow \int p(x_t|x_{t-1}u_{t-1})Bel(x_{t-1})dx_{t-1} \tag{1}$$

For new sensor data z_t the state estimation is updated to the a posteriori state $Bel(x_t)$:

$$Bel(x_t) \longleftarrow \eta p(z_t|x_t)Bel^{-}(x_t) \tag{2}$$

To being able to perform the update step, the system must know how precise the sensor data and the prediction model (process model) are.

3.2 Rao-Blackwellized Particle Filter

Even though Kalman filter models linear processes optimally, it is not as useful for non linear ball modeling. That's why an extension for the Kalman filter is introduced. In RoboCup there can be the following non linear ball motions - as Fox and Kwok [14][15] decribed.
Main reasons for a non linear ball motion are:

1. Non linear motions of the observer. Because of traction loss and collisions [11][16] with other robots and obstacles the motion data of the robot can be very noisy.
   This makes it hard to distinguish between robot motions and motions of the ball.
2. Physical interaction of the ball with its environment. The ball is often bouncing off other robots, objects or the field border (which was removed just in 2005 in the RoboCup legged league), or it is being kicked by other robots.
3. Physical interaction of the ball with the observer. In cases in which the robot catches or kicks the ball, the ball position is highly dependent on the agent's actions

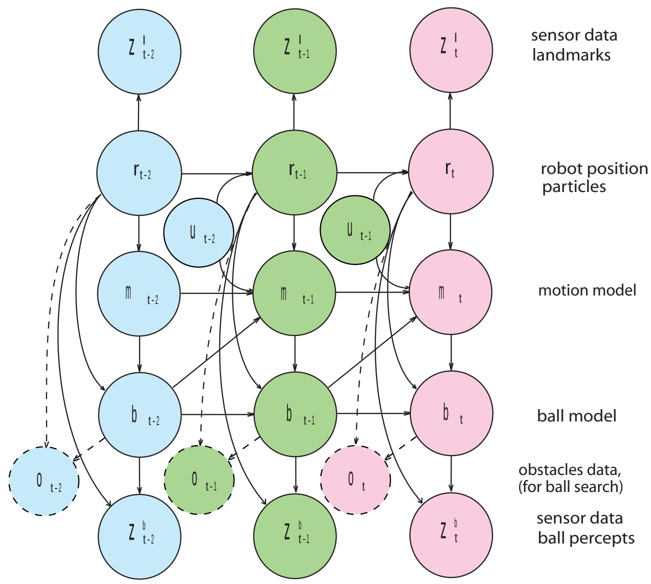

Fig. 1. *Bayesian network with motion model and obstacles data. Arrows represent direct dependencies between state dimensions.*

3.3 The Bayesian Network for the RBPF

The following section should show the dependencies of the state dimensions as well as the dependencies of the state dimensions and the sensor data. Therefore those dependencies are described in a Bayesian network (fig. 1).

In the Bayesian network in fig. 1 the robot position has to be modeled right at the beginning. It determines the probability of detecting a landmark. The robot position r_t at time t results from the preceding robot position r_{t-1} as well as from the executed motion u_{t-1} at time $t-1$. The probability $p(z_t^b)$ for perceiving a ball is dependent from the motion model of the ball m_t (bounced, kicked, grabbed), the ball position b_t and the robot position r_t.

Thus the likelihood for the i. localization particle $r_t^{(i)}$ at time t is being calculated as follows (Bayes):

$$r_t^{(i)} \backsim p(r_t | r_{t-1}^{(i)}, m_{t-1}^{(i)}, b_{t-1}^{(i)}, z_t, u_{t-1}) \tag{3}$$

The motion model $m_t^{(i)}$ is being calculated as:

$$m_t^{(i)} \backsim p(m_t | r_t^{(i)}, m_{t-1}^{(i)}, b_{t-1}^{(i)}, z_t, u_{t-1}) \tag{4}$$

The calculation of the ball position and speed $b_t^{(i)}$:

$$b_t^{(i)} \backsim p(b_t | r_t^{(i)}, m_{t-1}^{(i)}, b_{t-1}^{(i)}, z_t) \tag{5}$$

a) b)

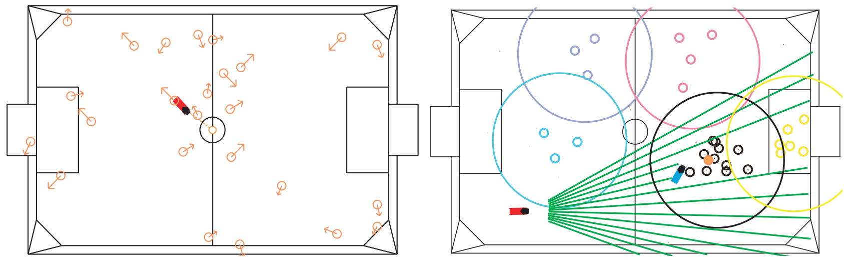

Fig. 2. *a) Particle distribution after the ball was lost. The particles are equally distributed over the field, b) after some executed search motions, the particles converge, the green lines in front of the left robot represent free areas in the field of view of the robot.*

3.4 Negative Information for Object Modeling

If an object cannot be seen for a long time it is considered to be lost and the robot starts with searching motions. In the example of a robot looking for a ball, the robot switches its particle representation from egocentric to allocentric. One advantage is that odometry errors doesn't effect the propagation of allocentically modeled particles. Then the robot can start scanning the field systematically.

While searching the ball, the robot uses negative information. This means that in areas in which the ball was expected to be but where it couldn't be seen, the hypotheses are erased, causing a higher position probability for the ball in areas where the robot hasn't searched for a while. By using obstacles information, false negative information, caused by robots on the field and thus occluding the ball, can effectively be avoided.

Obstacles Information The obstacles recognition was developed for being able to model robots and other objects on the field e.g. to avoid collisions while moving on the field. If this information is used for the ball search the robot can much better determine its field of view and differenciate between visible and occluded areas (fig. 3 a). The obstacle recognition is performed by scanning the green area in front of the robot as it is expected to be free. If the green area ends it gives hint for an obstacle.

In the hypothesis space only those particles are updated, which weren't occluded by obstacles an thus should have been directly visible. All remaining hypotheses are not updated because there is no sensor information available, as fig. 4 shows.

3.5 Estimation of the Ball Position

The ball position is calculated from the particle distribution by clustering. With clustering different areas of the hypothesis space with a high particle density can

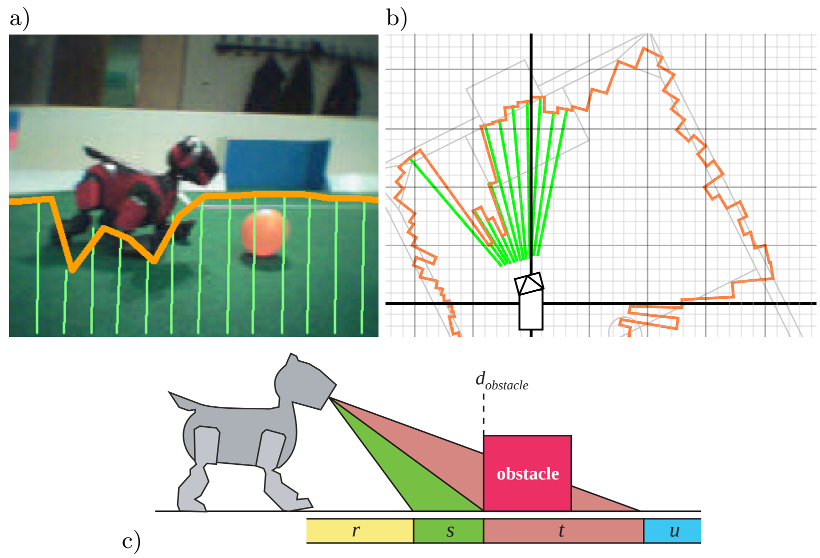

Fig. 3. *a) Free space in front of the robot, the ball is not recognized as an object, b) the green lines show how far the free area in front of the robot goes, c) the main different vision areas for a robot detecting an obstacles.*

be found and compared to each other. The cluster with the most particles determines the estimated object position. We tried different methods of clustering for position estimation. One method is to subdivide the hypothesis space with a grid [15], another method which was used here is the agglomerative clustering [9] which doesn't need a grid.

4 Experiments

We conducted an experiment to point out advantages of the introduced RBPF compared to an usual Kalman filter as it is often used in the RoboCup.

In this experiment a ball rolls towards the field border and bounces. This experiment was performed with an usual Kalman filter and with a RBPF. Results showed that the Kalman filter needed about two seconds to model the new velocity accurately after the ball bounced off the field border. This is due to the linearity expectations the Kalman filter, propagating the ball position into the border (fig. 5 a).

The RBPF was able to recalculate the new ball speed much quicker (fig. 5 b), even though self localization was not perfect $\sigma_{pos} = 100mm$.

5 Conclusion

A new object modeling approach, using RBPF in combination with negative and obstacles information was be presented. It was shown that this approach leads

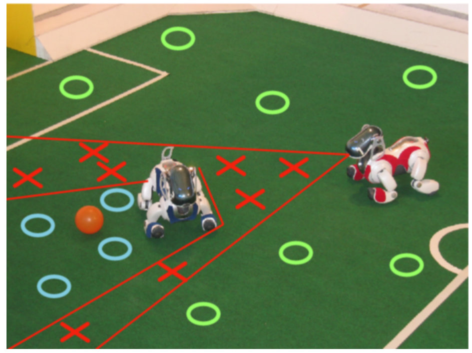

Fig. 4. *Example for hypothesis update: Only those hypotheses, directly visible to the robot are updated and erased from the hypothesis space (crosses). Hypotheses outside of the field of view, or behind an obstacle remain untouched (O's).*

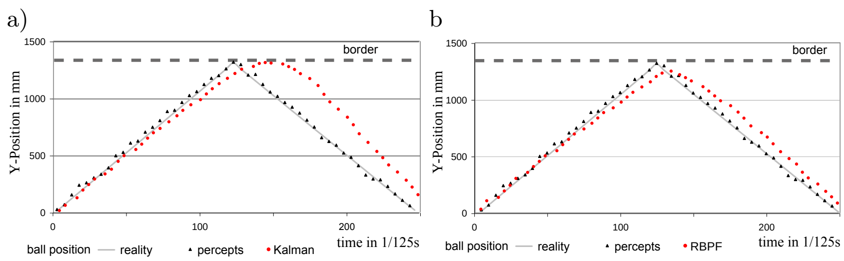

Fig. 5. *Exp.: a) The Kalman filter cannot react quickly enough to the direction change of the ball due to its linearity assumption, b) the RBPF models the deflection explicitly, enabling it to react quickly.*

to better results compared to a Kalman filter dealing with non linearities. The separation between self localization and object tracking could be overcome as well. In another experiment it was shown, that the use of negative information compared with obstacles information let the robot find a lost object quicker, than by arbitrary search. An open task is the modeling of other agents e.g. another robot is kicking a ball, because the player's model seems to be quite inefficient today. Future work should cope with multi agent and cooperative environment modeling, enabling the robots to execute plans on a common and consistent world model.

Even though the presented algorithm was introduced in a special domain, the results can be applied to a variety of other areas as well, as no special sensors or architecture are necessary.

References

1. C. Andrieu and A. Doucet. Particle filtering for partially observed gaussian state space models. 2002.

2. R. T. Comittee. Sony four legged robot football league rule book.
3. F. Dellaert, W. Burgard, D. Fox, and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 1999.
4. F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots.
5. A. Doucet, N. de Freitas, K. Murphy, and S. Russell. Rao-blackwellised particle filtering for dynamic bayesian networks.
6. D. Fox. Adapting the sample size in particle filters through kld-sampling. *International Journal of Robotic Research (IJRR)*, 2003.
7. D. Fox and W. Burgard. Active markov localization for mobile robots. 1998.
8. D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *In, Proc. of the National Conference on Artificial Intelligence*, 1999.
9. K. Fukunaga. In *Introduction to Statistical Pattern Recognition*, 1990.
10. J.-S. Gutmann and D. Fox. An experimental comparison of localization methods continued. *IROS 2002*, 2002.
11. J. Hoffman and D. Göhring. Sensor-actuator-comparison as a basis for collision detection for a quadruped robot. In *VIII. RoboCup Symposium 2004*, 2004.
12. Z. Khan, T. Balch, and F. Dellaert. A rao" blackwellized particle filter for eigentracking.
13. C. Kwok and D. Fox. Map-based multiple model tracking of a moving object. In *VIII. RoboCup Symposium, 2004*, 2004.
14. C. Kwok and D. Fox. Map-based multiple model tracking of a moving object. 2004.
15. C. T. Kwok. Robust real-time perception for mobile robots, dissertation thesis. 2004.
16. M. J. Quinlan, C. L. Murch, R. H. Middleton, and S. K. Chalup. Traction monitoring for collision detection with legged robots. In *VII. RoboCup Symposium 2003*, 2003.
17. T. Röfer and M. Jüngel. Vision-based fast and reactive monte-carlo localization. *ICRA 2003*, 2003.
18. S. Thrun, D. Fox, F. Dellaert, and W. Burgard. Particle filters for mobile robot localization.