# Exploiting the Unexpected: Negative Evidence Modeling and Proprioceptive Motion Modeling for Improved Markov Localization

Jan Hoffmann, Michael Spranger, Daniel Göhring, and Matthias Jüngel

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

**Abstract.** This paper explores how sensor and motion modeling can be improved to better Markov localization by exploiting deviations from expected sensor readings. Proprioception is achieved by monitoring target and actual motions of robot joints. This provides information about whether or not an action was executed as desired, yielding a quality measure of the current odometry. Odometry is usually extremely prone to errors for legged robots, especially in dynamic environments where collisions are often unavoidable, due to the many degrees of freedom of the robot and the numerous possibilities of motion hindrance. A quality measure helps differentiate the periods of unhindered motion from periods where robot motion was impaired for whatever reason. Negative evidence is collected when a robot fails to detect a landmark that it expects to see. Therefore the gaze direction of the camera has to be modeled accordingly. This enables the robot to localize where it could not when only using landmarks. In the general localization task, the probability distribution converges more quickly when negative information is taken into account.

## 1 Introduction

Selflocalization, the estimation of position and orientation of a mobile robot, remains an important and valuable task for mobile robotics. One of the most successfully applied approaches is called *Monte-Carlo-Localization*. This method is used in numerous robot navigation problem domains, such as office navigation [1], museum tour guides [13], RoboCup [7], as well as outdoor or less structured environments [9]. We propose 2 extensions affecting the sensor model as well as the motion model.

1. We show how *negative information* can be incorporated into Monte Carlo localization. The sensor model is extended by modeling the probability of non-detection events.

A. Bredenfeld et al. (Eds.): RoboCup 2005, LNAI 4020, pp. 24–35, 2006.

2. The motion model is improved through careful modeling of proprioceptive information. The resulting model is incorporated into the action update of the particle filter.

The adjustments and changes presented improve the general ability to localize and also allow the robot to localize in areas where it was previously unable. They enable the robot to quickly recover its belief after collision events and to adjust quickly to large displacements (kidnapped robot).

Negative information denotes the ascertained absence of expected sensor readings. This is incorporated into the current belief much like an additional sensor. Proprioception is based on the comparison of actual motion to intended motion. This information is used to enhance the influence of action commands onto the belief. The extensions each prove to be a useful addition to the particle filter used, but are not particle filter specific approaches and can be used in other Bayes Filters. The positive impact on localization is shown in real world experiments using the Sony Aibo ERS-7 robot.

## 2 Monte Carlo Localization

The Monte Carlo Localization method is a probabilistic method, utilizing Bayes law and the Markov assumption. The robot maintains a set of samples, called particles. The particles approximate the belief of the robot's position, a probability distribution over the possible positions of the robot. The current belief of the robot's position is modeled as particle density, allowing for multi-modal probability distributions and beliefs. Each particle represents a hypothetical position of the robot. Belief $Bel(s_t)$, the localization estimate at time $t$, to be at position $s_t$ is determined by *all* previous robot actions $u_t$ and observations $z_t$. Using Bayes law and the Markov assumption, $Bel(s_t)$ can be written as a function that only depends on the previous belief $Bel(s_{t-1})$, the last robot action $u_{t-1}$, and the current observation $z_t$:

$$Bel^{-}(s_t) \longleftarrow \int \underbrace{p(s_t|s_{t-1}, u_{t-1})}_{\text{motion model}} Bel(s_{t-1}) ds_{t-1} \tag{1}$$

$$Bel(s_t) \longleftarrow \eta \underbrace{p(z_t|s_t)}_{\text{sensor model}} Bel^{-}(s_t) \tag{2}$$

with normalizing constant $\eta$. Equation 1 shows the *a priori* belief $Bel^{-}(s_t)$ which takes into account the previous belief and propagates it using the motion model of the robot. It is the belief prior to the measurement. The measurement is then incorporated into the belief as described in (2) using the sensor model ('sensor updating'). In Markov localization, given an initial belief $Bel(s_0)$ at $t = t_0$, the robot updates its belief using odometry and then incorporates new sensor information. Each time new information arrives the robot updates its particle distribution using the previous motion command, the resulting distribution is

updated using the gathered sensor information. This 2 step operation requires 2 models. The *motion model* $p(s_t|s_{t-1}, u_{t-1})$ tries to model the effect of motion commands on the hypothetical positions. The *sensor model* incorporates environment and sensor information regarding this environment into the current belief. The particle filter employed for our work is based on the method described in [10]. Here particles consist of a robot pose and a probability. The robot pose $(x, y, \theta)$ represents the position and orientation of the robot ($x$,$y$ coordinates on the field in mm and orientation in radians). The likelihood $p$ is a measure of the plausibility of the hypothesis being at the specified robot pose. The approach first moves all particles according to the motion model of the action chosen. Afterwards the probabilities of the particles are adjusted using the sensory input and the sensor model. In a third step, called *resampling* particles are moved, deleted from the particle set or injected from observation, based on their probability.

## 3 Proprioceptive Motion Modeling

If obstacle avoidance fails robots are often unable to detect collisions since many designs, like the robot used in this work, lack touch sensors or bumpers. Apart from the current action failing, collisions (and subsequently being stuck) have severe impact on the robot's localization if odometry is used to any degree in the localization process. For these approaches to be robust against collisions, they tend to not trust odometry much. This paper is based on work dealing with collisions detection for a Sony Aibo using the walking engine and software framework described in [3]. The approach uses the servo motor's direction sensors for the task of estimating the quality of the odometry data gathered by the walking engine. In analogy to biology we call it proprioception because intrinsic data of the leg sensors is used.

### 3.1 Motion Model

The *motion model* consists of consecutive acquired odometry data incorporated into the Belief, as well as a random error $\Delta_{error}$, which is related to the distance traveled and the angle rotated. Every particle is updated using the odometry offset accumulated since the last update.

$$pose_{\text{new}} = pose_{\text{old}} + \Delta_{\text{odometry}} + \Delta_{\text{error}} \tag{3}$$

Where $\Delta_{\text{error}}$ is defined as

$$\Delta_{\text{error}} = \begin{pmatrix} 0.1d \times \text{random}(-1 \ldots 1) \\ 0.02d \times \text{random}(-1 \ldots 1) \\ (0.002d + 0.2\alpha) \times \text{random}(-1 \ldots 1) \end{pmatrix} \tag{4}$$

### 3.2 Collision Detection

The Aibo is not equipped with sensors to directly perceive the contact with obstacles. We have shown ways of detecting collisions using the sensor readings

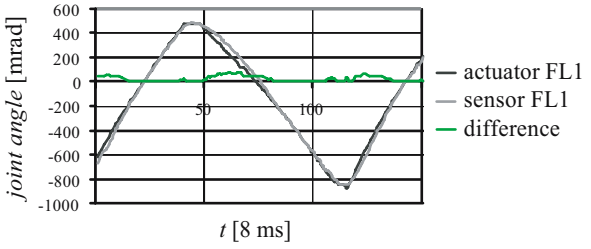



**Fig. 1.** Sensor and actuator data (shoulder joint FL1) for a freely walking robot. The corresponding difference function shows discrepancies between actuator and sensor data, caused by walking motions (peaks in the curve).

from the servo motors of the robot's legs in [3]. The comparison of motor commands and actual movement (as sensed by the servo's position sensor) can be used to detect collisions (see fig.1). This comparison has to compensate for the phase shift between the two signals and has to cope with arbitrary movements and accelerations produced by the behavioral layers of the robot. The method provides a *virtual collision sensor* that can be used to improve the motion model.

### 3.3 Extended Motion Model

The extended motion model accounts for the supplementary information provided by the collision detection module, by changing $\Delta_{\text{error}}$ as well as affecting the accumulated odometry update data in a random way. The binary decision of the collision sensor has a static impact on the motion noise. This means that $\Delta_{\text{error}}$ is no longer dependent on the distance traveled and the angle rotated, but rather is a uniform noise, within an interval expected to be a possible outcome of collisions. But also odometry data can not be fully relied upon, which is accounted for by randomly updating particles through the gathered odometry information, with the assumption that the robot most probably ends up somewhere between the requested destination and the starting point. The noise tries to account for the severe and unforeseeable impact of the collision. If collisions *are detected*, every particle is updated by:

$$pose_{\text{new}} = pose_{\text{old}} + \text{random}(0...1) \cdot \Delta_{\text{odometry}} + \Delta_{\text{error}}$$

Where $\Delta_{\text{error}}$ is

$$\Delta_{\text{error}} = \begin{pmatrix} 40 \times \text{random}(-1 \ldots 1) \\ 40 \times \text{random}(-1 \ldots 1) \\ 0.5 \times \text{random}(-1 \ldots 1) \end{pmatrix} \tag{5}$$

Otherwise, when no collision was detected, the motion model is not extended and the update is performed as usual(3). The effect of the changes are illustrated in fig.2 and 3.

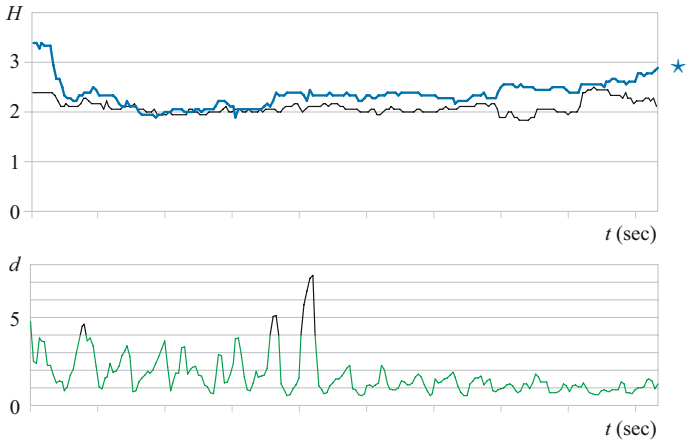



**Fig. 2.** Collision Experiment: A robot starts walking from the center circle in the direction of the goal and turns left before reaching the penalty area. The distributions entropy with ($\star$) and without the extended motion model (*top*) and the corresponding collision "sensor" (*bottom*, values greater than 4 are interpreted as a collision).

*Entropy.* We use the expected entropy $H$ as an information theoretical quality measure of the position estimate $Bel(s_t)$ [2]:

$$H_p(s_t) = -\sum_{s_t} Bel(s_t) \log(Bel(s_t)) \tag{6}$$

The sum runs over all possible states. The entropy of the particle distribution becomes zero if the robot is perfectly localized in one position. Maximal values of $H$ mean that $Bel(s_t)$ is uniformly distributed.

Fig. 3 illustrates the effect of the described motion modeling on the particle distribution. A robot is walking from the center circle in the direction of the goal when a collision occurs. It then continues towards the goal and turns left before reaching the penalty area. When the collision is modeled, the uncertainty in the belief is clearly visible and can be used to trigger appropriate robot behavior.

## 4 Negative Evidence Modeling

The used localization algorithm uses field lines and landmarks to perform the sensor updating of the current belief. We extend this approach through the use of *negative information*. Two main reasons make it hard to actually implement a system that integrates negative evidence: the target (landmark) may not be there or the sensor may simply be unable to detect the target (due to occlusions, sensor imperfections, imperfect image processing, etc.). Differentiating the two cases requires careful sensor modeling. We address this problem by considering the field of view of the robot and by using obstacle detection to estimate occlusions. Negative information has been used in object

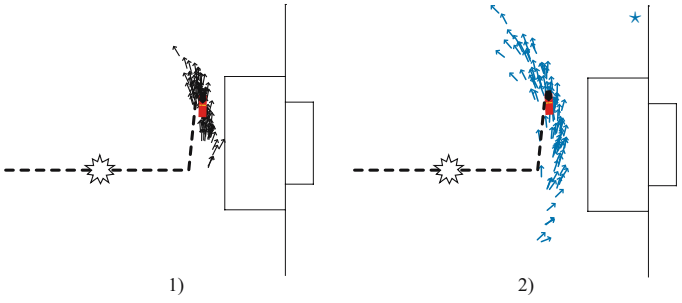



**Fig. 3.** Belief distribution without (2) and with (3) odometry quality used after a collision (marked by the star on the robot's path)

tracking (see [12] for an introduction and [5] for an overview). Not seeing the ball on the RoboCup field causes Monte Carlo particles to be deleted in that particular region [6]; occlusions are considered by explicitly modeling other robots' positions.

### 4.1 Sensor Model

Whenever the robot senses a landmark, the localization estimate is updated using the sensor model. This sensor model is acquired before the actual run. It describes the probability of the measurement $z$ given a state $s$ (position, orientation, etc.) of the robot. If no landmark is detected, the state estimation is updated using (only) the motion model of the robot. Sensory input in our case is measured bearings to landmark. The approach has been extended by accounting for field lines and edges [11] which require a slightly different model and method. The probability of the particles is derived from the measured bearings to landmarks.

### 4.2 Negative Information

Negative information describes the absence of a sensor reading in a situation where a sensor reading is expected given the current position estimate. To integrate negative information, imagine a binary sensor being added that fires whenever the primary sensor *does not* detect a particular landmark $l$. Its probability of it firing is given by:

$$p(z^{\star}_{l,t}|s_t) \tag{7}$$

This sensor model can be used to update the robot's belief whenever it fails to detect a landmark, i.e. when negative evidence is acquired. This rather coarse way of incorporating negative information can be refined by taking into account the range $r_t$ of the robot's sensors and possible occlusions $o_t$ of landmarks. The sensing range is the physical volume that the sensor is monitoring. In case of a stationary robot, $r_t = r_0$ is constant, for a mobile robot with a pan-tilt camera it is not. By $o_t$ we denote a means of detecting the occurrence of occlusions. In practice, this can be calculated from a map of the environment, directly sensed

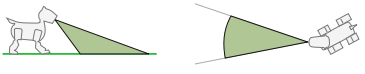
by a sensor such as a laser range finder, or derived from a model of moving objects in the environment. Combining the two yields the probability of not sensing an expected landmark $l$ is:

$$p(z^{\star}_{t,l}|s_t, r_t, o_t) \tag{8}$$

The absence of the detection of a landmark can be used in the sensor update step of the Iterative Bayesian Updating (see Algorithm 1).

### 4.3 Extended Sensor Model

**Field of View.** Here we show the camera of the Sony Aibo ERS-7. The ERS-7 is a legged robot with a camera mounted in its head. The camera has a horizontal opening angle of 55° and the robot's head has 3 degrees of freedom (neck tilt, head pan, head tilt). We abbreviate gaze direction by $\varphi = (\varphi_{\text{tilt1}}, \varphi_{\text{pan}}, \varphi_{\text{tilt2}})$. The sensing range is calculated by considering the field of view (FOV) of the robot:



**Occlusion.** In order to account for occlusions, we opted for an approach that has been used successful for detecting obstacles, referred to as 'visual sonar' [4, 8]: The camera image is scanned in vertical scan lines and unoccupied space in the plane of the field is detected, since it can only be of green or white color (field lines). Scanning for these colors tells the robot where obstacles are and where there is free space, which in turn can be used to determine whether the visibility of the landmark was impaired, i.e. if it was occluded by another robot or some other obstacle. More specifically, if the expected landmark lies in an area where the robot has detected free space, the likelihood of the corresponding pose estimate is decreased. If it lies outside of the detected free space, no information can be inferred. Taking FOV and occlusion into account, the sensor model for not perceiving an expected landmark is given by:

$$p(z^{\star}_{t}|s_t, z_{t,\text{obstacle}}) \tag{9}$$

Where $s_t = (x_t, y_t, \vartheta_t, \varphi_t)$ describes the robot state that consists of the *robot pose* (position $x_t, y_t$, and orientation $\vartheta_t$) and the current gaze direction $\varphi_t$. These are used to calculate the field of view of the camera. The sensor updating part of the localization code was adjusted to account for FOV and occlusion as described above. This means that sensor updating is triggered whenever there is a new camera image regardless of whether or not there was a percept. Before re-sampling, the weight of an individual particle is calculated as follows: Of all landmarks $L$, a subset of landmarks $L'$ is detected, the subset $L^{\star}$ is expected but not detected, and lastly the subset $L^{\diamond}$ is not detected, but was also not expected,

**Algorithm 1.** Iterative Bayesian updating incorporating negative evidence

1: $Bel^{-}(s_t) \longleftarrow \int p(s_t|s_{t-1}, u_{t-1})Bel(s_{t-1})ds_{t-1}$
2: **if** (landmark $l$ was detected) **then**
3: $\quad Bel(s_t) \longleftarrow \eta p(z_t|s_t)Bel^{-}(s_t)$
4: **else**
5: $\quad Bel(s_t) \longleftarrow \eta p(z^{\star}_{t,l}|s_t, r_t, o_t)Bel^{-}(s_t)$
6: **end if**

$L = L' \cup L^{\star} \cup L^{\diamond}$ and $L^{\star} \cap L' = \emptyset$. The probability of a particle $p_i$ is calculated by multiplying all the gathered evidences:

$$p_i = \underbrace{\prod_{l \in L'} s_l(\alpha_{\text{measd}}, \alpha_{\text{expd}})}_{\text{detected}} \cdot \underbrace{\prod_{l \in L^{\star}} s^{\star}_l(\varphi, \alpha_{\text{expd}})}_{\text{expected and not detected}} \tag{10}$$

The function $s_l$ is an approximation of the sensor model and returns the likelihood of sensing the landmark $l$ at angle $\alpha_{\text{measd}}$ for a particle $p_i$ that expects this landmark to be at $\alpha_{\text{expd}}$. Function $s^{\star}_l$ models the probability of not sensing the expected landmark $l^{\star}$ given the current sensing range as determined by $\varphi$, the robot pose associated with $p_i$, and the obstacle percept $z_{\text{obstacle}}$.

### 4.4 Experimental Results

*Localization Experiment.* The following experiment is a localization task on the real robot. The robot is placed on the field in front of a landmark facing outwards. The robot performs a scanning motion with its head (pan range $[-45^{\text{o}}, 45^{\text{o}}]$) but does not move otherwise. From where it is standing, it can only see one landmark. A panorama composed of actual robot camera images is shown in fig. 4. The *a priori* belief is assumed uniform. This position was chosen because it is a particulary difficult spot for the robot to localize given the limited sensor information. The bearing $\alpha_l$ to a landmark is used for localization because the distance measurement is prone to errors. Using just the bearing, only the orientation of the robot can be inferred.

In the following paragraphs, the basic localization excluding the use of negative information and localization incorporating negative information are compared. We will first give a more qualitative analysis of the particle distribution and then show how the entropy of the distribution decreases when negative information is considered.

*Particle Distribution.* The basic experiment was conducted using 100 particles for Monte Carlo localization. It was repeated on a log file (containing camera images and robot joint angles) using an increased particle count of 2000. This was done to reveal artifacts due to the small number of particles used in the standard implementation.
*Not using negative information.* Without using negative information, the robot is unable to localize (fig. 5). Only the orientation of the particles is adjusted

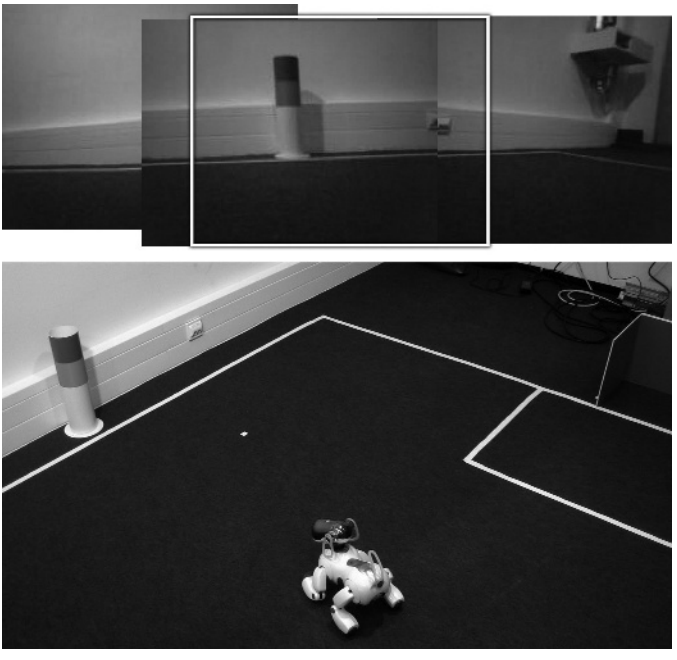


**Fig. 4.** *Top:* A panoramic view generated from actual camera images, single camera image highlighted. The robot can only see *one* landmark. *Below:* Photo of the experimental setup in our lab.

according to the sensor readings. The apparent clustering in fig. 5 is not stable and even after considerable time, the particles do not converge. The distribution for the larger sample set is uniform (w.r.t. position). Note that the distribution is not circular because the distance to the landmark was not used. Instead, only the bearing to the landmark was used. This results in a radial distribution resembling magnetic field lines.
*Incorporating negative information.* The negative information gained in this experiment is only seeing one landmark within the pan range. Incorporating this information, the robot is able to localize quickly. On average, the robot is reasonably well localized after 5-10 secs with a pose error of less than $\Delta p = (35\text{ cm}, 35\text{ cm}, 20^{\text{o}})$.

*Entropy.* We now consider the entropy of the particle distribution as defined earlier. Fig. 6 shows the progression of the distribution's entropy over time for the above localization experiment calculated from the 100 particle distribution.

*Not using negative information.* The experiments starts with a uniform particle distribution which equals to maximum entropy. When the landmark comes into view, a decrease in entropy is observed. This information gain is caused by the robot now knowing its relative orientation w.r.t. the landmark. Since there are

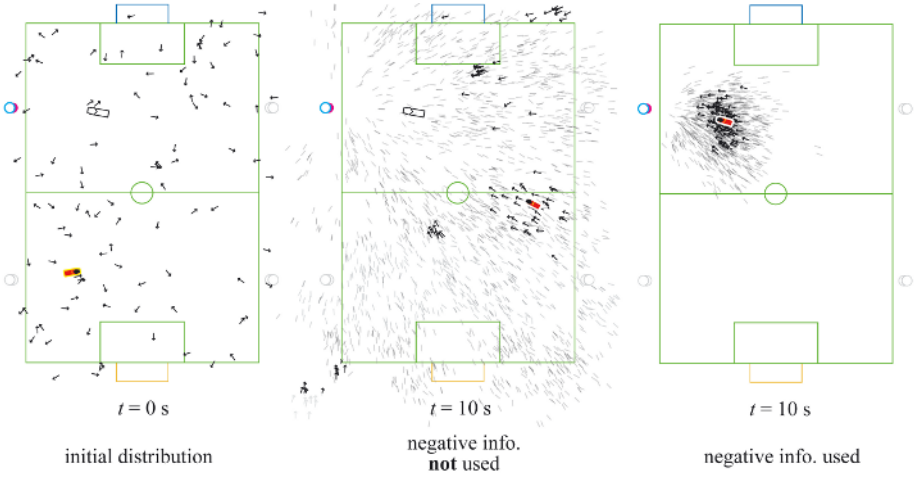



**Fig. 5.** Particle distribution not using negative information, initial uniform distribution and distribution after 10s. Solid arrows indicate Monte Carlo particles (100). The experiment was repeated using 2000 particles (shaded lines) to better represent the actual probability distribution. The actual robot position is indicated by the white symbol, the calculated robot pose by the solid symbol. Not using negative information and only using the bearing to the landmark the robot is unable to localize. Some clusters of particles form but they do not converge. As one would expect, the position distribution is almost uniform but the relative angle is quite distinct. *Right.* Particle distribution when negative information is incorporated, enabling the robot to localize quickly.

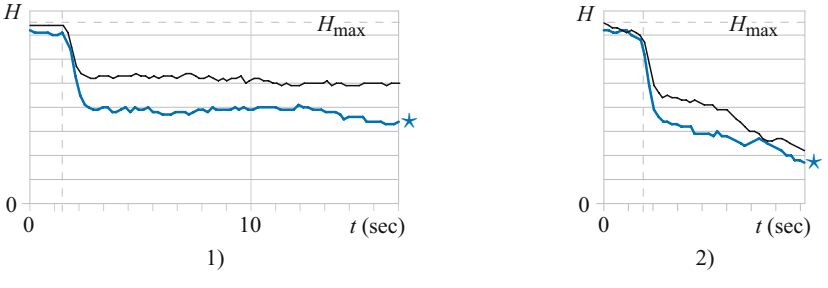



**Fig. 6.** Expected entropy of the belief in the localization task with (⋆) and without (thin line) using negative information. 1) At first the robot does not see the landmark. As soon as the landmark comes into the robot's view (indicated by the dashed vertical line), the entropy drops. Using negative information, the quality of the localization is greatly improved and the entropy continues to decrease over time. Note that the entropy decreases even before the landmark has been seen for the first time. 2) Additionally using field lines for localization enables the robot to localize. Incorporating negative information, the rate of convergence is higher and the entropy is significantly lower than without using it.

no constraints on the robot's position, the entropy remains at a relatively high level. Note that even though there is a drop in entropy, the localization estimate itself is still highly uncertain.
*Incorporating negative information.* When using negative information, the entropy decreases even before the first sensor reading. The information gain is much smaller than that caused by perceiving a landmark, but nevertheless noticeable. As soon as there is a percept, the negative information in combination with the knowledge of the robot's orientation results in a quick convergence of the particle distribution towards the actual robot pose. Remember that without using negative information no localization was observed.
*Using field lines for localization.* In our last experiment, field lines were used for localization in addition to landmarks. This enables the robot to localize quickly at the actual robot pose even when using the basic localization. Adding negative information, however, greatly increases the rate of convergence and the overall level of entropy is reduced even further. It is noteworthy that the decrease of entropy when incorporating negative information is not obscured by the usage of lines for localization (which offer a much higher information content than negative information).

## 5 Conclusion

We have demonstrated the power of integrating negative information as well as information about collisions into Markov localization.

We have shown how an odometry-based motion model can be improved using the knowledge about collisions with obstacles. This knowledge has been obtained by comparing the motor commands and the sensor readings of the leg joints. In the case of a collision the influence of the odometry on the motion model was reduced and extra noise was added that models the impact of an obstacle.

Incorporating negative information into the sensor model makes localization more stable even in areas where landmarks are rarely visible. The usage of negative information does, however, require very careful modeling. To avoid false negatives, the model needs to take into account the sensor's sensing range and possible occlusions of landmarks. We have presented how such modeling can be achieved for a Sony Aibo robot in the RoboCup environment. In real robot experiments, we have shown that using negative information, a robot is able to localize in positions where it otherwise would not. The entropy of the distribution is greatly reduced when negative information is incorporated and the rate of convergence towards the estimated position is increased. Future work will focus on how negative information can be used for other types of landmarks (e.g. field lines) and other sensors. Performance evaluation will be continued in more complex situations and the possibilities of reducing the number of particles necessary for robust Monte Carlo localization will be investigated.

## Acknowledgments

Program code used was developed by the GermanTeam, a joint effort of the Humboldt University of Berlin, University of Bremen, University of Dortmund, and the Technical University of Darmstadt. Source code is available for download at http://www.germanteam.org.

## References

1. D. Fox, W. Burgard, F. Dellart, and S. Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In *Proc. of AAAI*, 1999.
2. D. Fox, W. Burgard, and S. Thrun. Active Markov Localization for Mobile Robots. In *Robotics and Autonomous Systems*, 1998.
3. J. Hoffmann and D. Göhring. Sensor-Actuator-Comparison as a Basis for Collision Detection for a Quadruped Robot. In *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence. Springer, 2005.
4. J. Hoffmann, M. Jüngel, and M. Lötzsch. A Vision Based System for Goal-Directed Obstacle Avoidance. In *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence. Springer, 2005.
5. W. Koch. On Negative Information in Tracking and Sensor Data Fusion. In *Proceedings of the Seventh International Conference on Information Fusion*, pages 91–98, 2004.
6. C. Kwok and D. Fox. Map-based Multiple Model Tracking of a Moving Object. In *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence. Springer, 2005.
7. S. Lenser, J. Bruce, and M. Veloso. CMPack: A Complete Software System for Autonomous Legged Soccer Robots. In *AGENTS '01: Proceedings of the fifth international conference on Autonomous agents*, pages 204–211. ACM Press, 2001.
8. S. Lenser and M. Veloso. Visual Sonar: Fast Obstacle Avoidance Using Monocular Vision. In *Proceedings of IROS'03*, 2003.
9. M. Montemerlo and S. Thrun. Simultaneous Localization and Mapping with Unknown Data Association Using FastSLAM. 2003.
10. T. Röfer and M. Jüngel. Vision-Based Fast and Reactive Monte-Carlo Localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-2003), Taipei, Taiwan*, pages 856–861, 2003.
11. T. Röfer and M. Jüngel. Fast and robust edge-based localization in the sony four-legged robot league. In *7th International Workshop on RoboCup 2003 (Robot World Cup Soccer Games and Conferences)*, Lecture Notes in Artificial Intelligence. Springer, 2004.
12. S. Särkkä, T. Tamminen, A. Vehtari, and J. Lampinen. Probabilistic Methods in Multiple Target Tracking, Research Report B36. Technical report, Laboratory of Computational Engineering Helsinki University of Technology, 2004.
13. S. Thrun, D. Fox, and W. Burgard. Monte Carlo Localization with Mixture Proposal Distribution. In *Proc. of the National Conference on Artificial Intelligence*, pages 859–865, 2000.