Negative Information and Proprioception in Monte Carlo Self-Localization for a 4-Legged Robots

Jan Hoffmann, Michael Spranger, Daniel Göhring, and Matthias Jüngel
Humboldt-Universität zu Berlin, Institut für Informatik,
LFG Künstliche Intelligenz, Unter den Linden 6,
10099 Berlin, Germany, http://www.aiboteamhumboldt.com

Abstract

This work explores how extended modeling of sensors and robot motion can be used to improve Markov localization by monitoring deviations of actual measurements from expected sensor readings. By comparing target and actual motions of robot joints, *proprioception* is achieved yielding a quality measure for the current odometry. a quality measure for odometry that helps differentiate periods of unhindered motion from periods where robot motion was impaired for whatever reason. By *negative information* we denote the absence of an expected sensor reading. Negative information is seldom used in localization because it yields less information than positive information (i.e. sensing a landmark) and a sensor often fails to detect a landmark, even if it falls within its sensing range. We address these difficulties by carefully modeling the sensor to avoid false negatives. In real world experiments, we are able to demonstrate that a robot is able to localize in positions where without the use of negative information it could not.

1 Introduction

The estimation of position and orientation of a mobile robot is a cruical task in mobile robotics. One of the most successfully applied approaches is called *Monte-Carlo-Localization*. This method is used in numerous robot navigation applications, such as office navigation [2], museum tour guides [19], RoboCup [11] [**?**], as well as outdoor or less structured environments [13]. 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*.
2. The motion model is improved by modeling of proprioceptive information. The resulting model is incorporated into the action update of the particle filter.

The presented adjustments and changes improve the general ability to localize and also allow the robot to localize in areas where it was previously unable to do so. 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.

This work is motivated by the desire to improve the localization of robots that compete in the *RoboCup Sony Four Legged League*. In this league, two teams of four robots compete on a field of green carpet sized 6 m x 4 m. There are two colored goals and white field lines which define the dimensions of the field. There is also a center line, a center circle and penalty areas near each goal. To help the robots localize there are four cylindrical landmarks at the side of the field. These beacons have a simple two color code that uniquely identifies them. The Aibo robot itself has a camera with a field of view of 55^o and a resolution of only 208×160 pixels YUV. It is built into the robot's head which has 3 degrees of freedom. The robot's legs have 3 degrees of freedom each. Due to their small size and low power requirements the robots have rather limited computational power (576 MHz processor). This somewhat limits the sensory capabilities of the Aibos compared to robots that are equipped with laser range finders, sonars and a possibly high end notebook and requires for efficient algorithms and attention control. The nature of the soccer games in *RoboCup Sony Four Legged League* makes the localization task even more challenging. The robots have little evidence whether desired movements were successful or not. Odometry data is of poor quality as the robots often slip on the ground or run into each other. Furthermore, the robots are required to track the ball which makes localization even more difficult as landmarks are only seen infrequently and may be occluded by other robots. In the following sections we will present ways to address these challenges.

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. The 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:

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

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

with normalizing constant η. 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 [16]. Here particles consist of a robot pose and a probability $(x, y, \theta), p$. The robot pose (x, y, θ) 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.

The RoboCup uses a color coded environment. The distance and bearing to landmarks and the goals are used for sensor update. Other features of the domain are field lines which are also used by some approaches [16]. Goals and landmarks are identified by the camera located in the robot's head. The color pattern of the features is used to identify landmarks. The sensory input of the leg and head joints is used to determine gaze direction, field of view, as well as the direction of identified features. The motion model is usually determined before the game by measuring the effect of motion commands on the actual displacement of the robot (see next section).

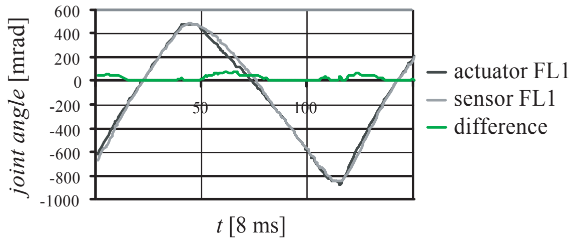


Figure 1: Sensor and actuator data (shoulder joint FL1) for a freely walking robot. The corresponding difference function shows discrepancies between actuator and sensor data, caused by walking motions (peaks in the curve).

3 Proprioceptive Motion Modeling

Many research efforts in mobile robotics aim at enabling the robot to safely and robustly navigate and to move about both known and unknown environments (e.g. the rescue scenarios in the RoboCup Rescue League, planetary surfaces [21]). While wheeled robots are widely used in environments where the robot can move on flat, even surfaces (such as office environments or environments that are accessible to wheelchairs [10]), legged robots are generally believed to be able to deal with a wider range of environments and surfaces. There are many designs of legged robots varying in the number of legs used, ranging from insectoid or arachnoid with 6, 8 or more legs (e.g. [1]), 4-legged such as the Sony Aibo [4], humanoid: 2-legged (e.g. [14]).

Obstacle avoidance is often realized using a dedicated (360°) range sensor [20]. Utilizing vision rather than a dedicated sensor is generally a much harder task since a degree of image understanding is necessary. For the special case of color coded environments, straight forward solutions exist that make use of the knowledge about the robot's environment (such as the color of the surface or the color of obstacles [12], see also previous section). If, however, 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. Such robots run into walls and continue to do so since they have no way of telling that they are in a fatal situation. Apart from the current action failing (e.g. the target position not being reached), collisions and subsequently being stuck have severe impact on the robot's localization. This is due to the motion update step in Bayesian filtering where the current motion of the robot is incorporated into its belief (cf. 1). This updating is usually limited to incorporating the robot's own motion which is commonly referred to as odometry. While calculation of odometry is straightforward in a wheeled robot (counting turns of the wheels), the task is much more complex for a legged robot. Forward kinematic can be used to a certain extent [15], but this requires well defined gait patterns. Since gait optimization is often done using genetic optimization, patterns tend to be highly complex and a physical simulation of the robot would be necessary to adequately predict its motion. Such gaits require calibration for them to be used in actual robotic applications [**?**]. However well the odometry is calibrated, robot locomotion remains a stochas-

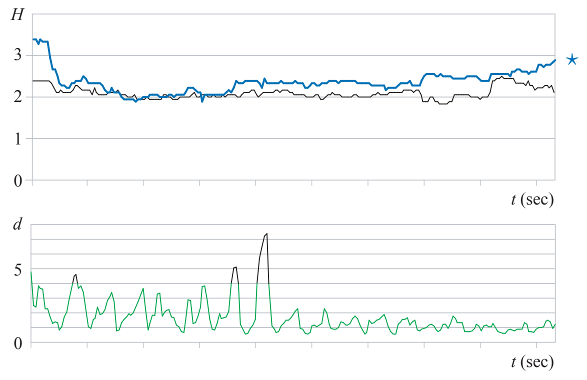


Figure 2: Bottom: The *collision sensor* - values greater than 4 are interpreted as a collision. Top: The entropy of the belief (represented by the sample set) with and without (thin line) improving the motion model. When the enhanced model is used, the entropy increases during collisions, because noise is added to the distribution.

tic process and is never quite reproducible. In the RoboCup domain, there is an additional source of errors: other robots competing for the ball. Robots often push each other or interlock their legs causing motions to have erratic outcome. The following approach is based on work dealing with collisions detection for a Sony Aibo using the walking engine and software framework described in [**?**]. 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 this *proprioception* because internal sensors are used to determine the state of the robot's body.

3.1 Motion Model

The *motion model* consists of consecutive acquired odometry data incorporated into the belief, as well as a random error Δ_{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 Δ_{error} is defined as

$$\Delta_{\text{error}} = \begin{pmatrix} 0.1d \times \text{random}(-1\ldots1) \\ 0.02d \times \text{random}(-1\ldots1) \\ (0.002d + 0.2\alpha) \times \text{random}(-1\ldots1) \end{pmatrix} \tag{4}$$

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 from the servo motors of the robot's legs in [**?**]. 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.

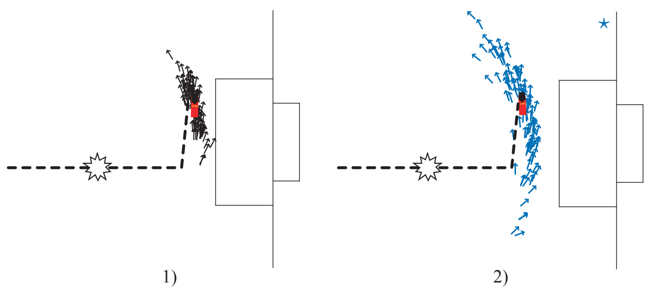


Figure 3: Belief distribution without (1) and with (2) odometry quality used after a collision (marked by the star on the robot's path).

3.3 Extended Motion Model

The extended motion model accounts for the supplementary information provided by the collision detection module, by changing Δ_{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 Δ_{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 Δ_{error} is

$$\Delta_{\text{error}} = \begin{pmatrix} 40 \times \text{random}(-1\ldots1) \\ 40 \times \text{random}(-1\ldots1) \\ 0.5 \times \text{random}(-1\ldots1) \end{pmatrix} \tag{5}$$

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 can be seen in fig.2 and 3.

Entropy We use the expected entropy H as an information theoretical quality measure of the position estimate $Bel(s_t)$ [3]:

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

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 continous 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.

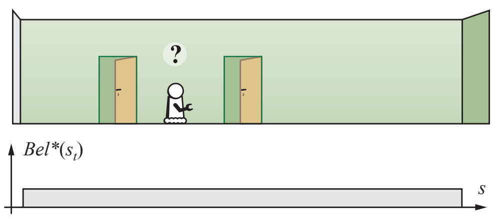


Figure 4a: ($t = t_0$) Illustration of a robot localizing in an office hallway. The robot has a sensor to detect doors. At the beginning, the robot does not know its position in the hallway (uniform belief distribution $Bel^\star(s_t)$). At this time, no sensing of the world takes place.

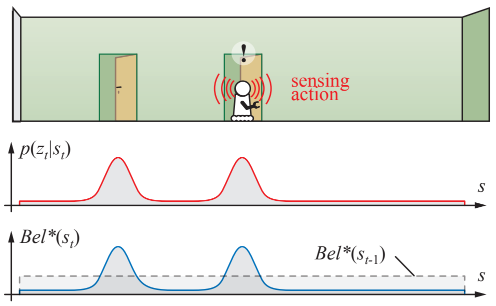


Figure 4b: ($t = t_1$) The robot has moved down the hallway and now senses a door $p(z_t|s_t)$ which results in the shown belief $Bel^\star(s_t)$. It has two peaks since the robot could be standing in front of either door. The previous distribution is illustrated by the dashed line.

4 Exploiting Negative Information

The classic example of negative information was described in the Sherlock Holmes case "Silver Blaze." In this case, a house has been broken into. Under such circumstances, one would expect the watch-dog to bark. The curious incident of the non-barking of the dog in the nighttime provides Holmes with the information that the dog must know the burglar, allowing him to solve the case. Applied to mobile robot localization, this means that conclusions can be drawn from expected but actually missing sensor measurements [7]. Markov localization methods, in particular Monte Carlo localization, have proven their power in numerous robot navigation tasks, e.g. in office environments [2], in the museum tour guide Minerva [19], in the highly dynamic RoboCup environment [11], and outdoor applications in less structured environments [13]; an evaluation of the various algorithmic approaches is given in [5].

Our work is focussed on localization based on landmarks. Whenever a 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. Sensor updates only occur when landmarks are detected. If no landmark is detected, the state estimation is updated using (only) the motion model of the robot.

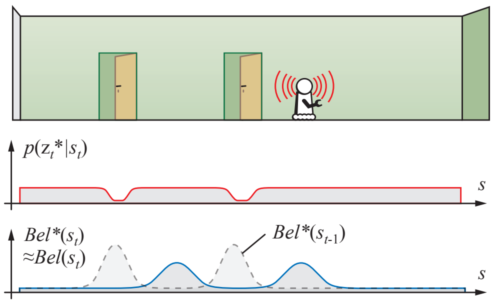


Figure 5a: ($t = t_3$) The robot moves on. There are no doors nearby so the "door sensor" does not sense a door. The sensor update distribution is shown in $p(z_t^\star|s_t)$. This negative information is of negligible use at this position: it does not help differentiate between the peaks.

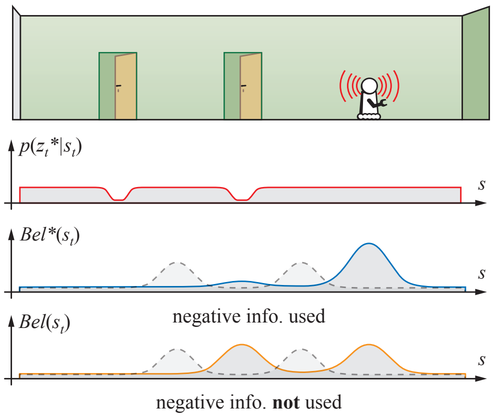


Figure 5b: ($t = t_4$) The robot moves on and the door sensor still does not sense a door. $Bel^\star(s_t)$ shows the belief if negative information is taken into account, whereas $Bel(s_t)$ shows the belief without using negative information to better illustrate the case. As can be seen from the diagram, using negative information allows the robot to rule out the left peak.

Example. Consider a robot driving down a corridor as shown in fig. 4a-5b. The robot has a sensor to detect doors when it is standing in front of one. Let us assume further that the robot is moving to the right but is oblivious of its starting position. As it starts to move to the right it passes and senses a door. Given this information, it could be standing in front of either of the doors (states s_{left} and s_{right}). As it moves on, it does not pass another door for some time. At time $t = t_3$, if s_{left} had been the true position, the robot would have had passed another door by now. Using the negative information of not perceiving a door, the belief based on s_{left} can be ruled out. As Thrun, Bugard, and Fox put it quite graphically, "not seeing the Eiffel Tower in Paris implies that it is unlikely that we are right next to it" [18].

We present a localization approach that incorporates such negative information. To our knowledge, no explicit study of using negative information in Markov localization has

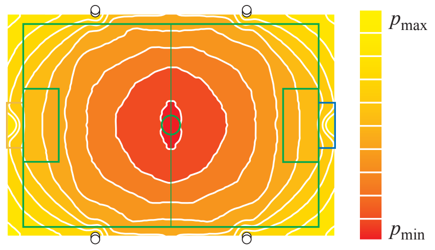


Figure 6: Probability of *not* sensing a landmark for a robot on a RoboCup soccer field. For a robot located around the center of the field, it is hard to miss landmarks.

been published. One difficulty is brought about by the fact that, generally speaking, sensing a landmark constitutes a greater information gain than not sensing one simply because there are many positions within the robot's environment from where the landmark cannot be perceived. A landmark is, by definition, something that stands out in an environment.

The other difficulty in implementing a system that uses negative information on a real robot is that there are two main reasons for the absence of an expected sensor reading: the target 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 is not a trivial task and 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 modeling has been applied to object tracking (see [17] for an introduction and [7] for an overview). The event of not detecting an object is treated as evidence that can be used to update its probability density function [8]. In the RoboCup domain, not seeing the ball on the field can be used to delete Monte Carlo particles in that region as long as occlusions are considered [9]. Negative information is also mentioned in the context of simultaneous localization and mapping (SLAM) where it is used to adjust the confidence in landmark candidates [13].

4.1 The Notion Of 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. Fig. 6 shows the probability $p(z^\star_t|x_t, y_t)$ of not sensing a landmark on a RoboCup field at position (x_t, y_t) summed over all possible robot orientations. This figure also shows that it is most likely for the robot to sense a landmark when it is standing in the middle of the field. The likelihood of not sensing a landmark is highest for positions at the edge of the field as the robot may be facing outwards.

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 $l$ detected) then
3:     $Bel(s_t) \longleftarrow \eta p(z_t|s_t) Bel^-(s_t)$
4: else
5:     $Bel(s_t) \longleftarrow \eta p(z^\star_{t,l}|s_t, r_t, o_t) Bel^-(s_t)$
6: end if
```

This rather coarse way of incorporating negative information can be refined by taking into account the sensing 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 whether or not occlusions have occurred. 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:

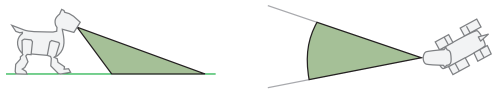
$$p(z^\star_{t,l}|s_t, r_t, o_t) \tag{8}$$

Whenever a landmark is not detected, it can be used in the sensor update step of the Iterative Bayesian Updating (see Algorithm 1).

4.2 Sensor Modeling For The Sony Aibo

Field of View

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 successfully for detecting obstacles, referred to as 'visual sonar' [6; 12]: 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 if the visibility of the landmark is impaired, i.e. if it is occluded by other 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 inference can be made.

Taking FOV and occlusion into account, the sensor model for not perceiving an expected landmark is given by:

$$p(z_t^\star | s_t, z_{t,\text{obs}}) \tag{9}$$

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 ϑ_t) and the current gaze direction φ_t.

4.3 Experimental Results

In the following experiments, unless otherwise stated, only landmarks were used for localization to emphasize the effect of using negative information.

Monte Carlo Localization, Implementation

This work is based on the Monte Carlo localization described in [16] which also serves as a base line implementation. Sensor updating was extended to account for FOV and occlusion as described. This also requires sensor updating to be triggered by new camera images 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, the 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: $L = L' \cup L^\star \cup L^\diamond$ and $L^\star \cap L' = \emptyset$. The probability of a particle p_i is calculated by multiplying all the likelihoods of all gathered evidences:

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

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

Preliminary Experiment

For illustration purposes, we conducted a preliminary experiment in simulation. In this experiment, the robot starts out being well localized and is then displaced to a position where it is not able to get any new sensor information (fig. 7). It is similar to the kidnapped robot problem, but here we emphasize the moment right after the robot is displaced rather than investigating how fast it can recover. The effect of the displacement on the Monte Carlo particle distribution is the following: particles which represent the previous belief become less likely when negative information is taken into account (i.e. the information that the landmark is not detected where it is expected). The distribution diverges towards particles which were less likely prior to the displacement. Particles representing the previous belief are eventually eliminated from the distribution because they are inconsistent with the current (negative) sensor data. Particles which differ from the previous belief just enough to be compatible with the current sensor data are favored; particles remain close to where the robot was last able to localize. This does, in most cases, better represent what has happened to the robot than distributing the particles uniformly over the entire field.

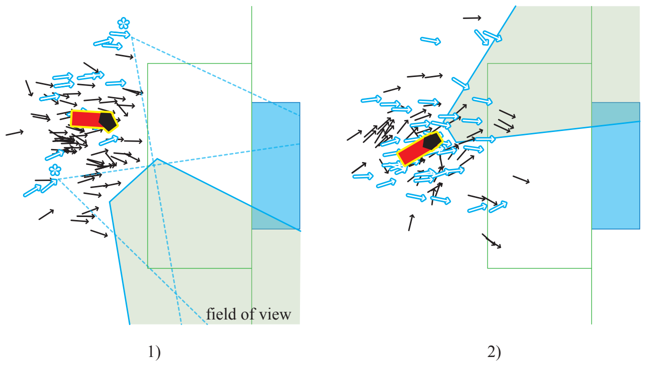


Figure 7: Incorporating negative information. White (outlined) arrows denote particles that receive negative information and that are therefore less likely than others. In (1), the effect of using negative information is shown for a robot that is well localized and frequently sees landmarks. (2) Distribution shortly after the robot has been displaced (kidnapped): particles facing the goal are less likely and will eventually be eliminated from the distribution.

4.4 Localization Experiment

The following experiment is a localization task using the real robot. The robot is placed on the field at the location indicated in fig. 9, facing outwards. The robot performs a scanning motion with its head (pan range $[-45^\circ, 45^\circ]$) but does not move otherwise. From its position, it can only see one landmark. A panorama composed of actual robot camera images is shown in fig. 8. 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. Two quantities can be used when a landmark is seen: its size in the camera image can be used to estimate the distance to the landmark d_l and the relative angle to the landmark (bearing, α_l) can be calculated from its position within the image. In practice we only use the bearing because the distance measurement is error prone. Using just the bearing, only the orientation of the robot can be inferred. Note that this differs from triangulation where distances are used.

In the following paragraphs, the basic localization not using negative information and localization incorporating negative information are compared. We first qualitatively analyze 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, robot joint angles, and odometry data using an increased particle count of 2000 to get a better representation of the probability distribution.

Not using negative information. Without using negative information, the robot is unable to localize (fig. 10). Only the orientation of the particles is adjusted according to the sensor readings. The apparent clustering in the small sample set in fig. 10 is not stable and, even after considerable time,

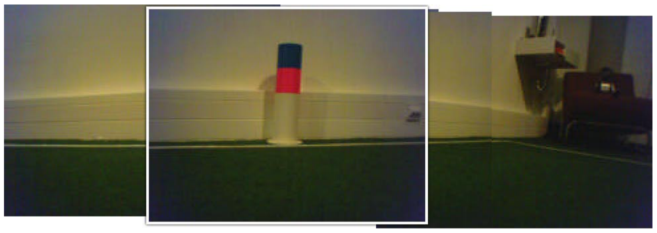


Figure 8: A panorama view generated from actual camera images, single camera image highlighted. The robot can only see *one* landmark.

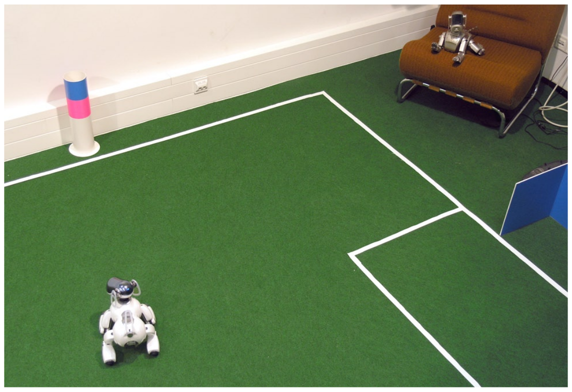


Figure 9: *Experimental setup:* Robot is standing at the position shown in the photo. It performs a scanning motion with its camera.

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 not seeing but one landmark within the pan range (pardon the double negation). Incorporating this information, the robot is able to localize quickly. On average, the robot is reasonably well localized after about 10 secs with a pose error of less than $\Delta p = (25\text{ cm}, 25\text{ cm}, 20^o)$.

Entropy

Entropy is considered for the localization task as defined in equation 6. Fig. 12 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 run 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 due to the robot being able to now infer its relative orientation w.r.t. the landmark. Since there are no constraints on the robot's position, the entropy remains at a relatively high level. This is easily seen by separately calculating the entropy of the angle and position distributions. Note that even though there is a drop in

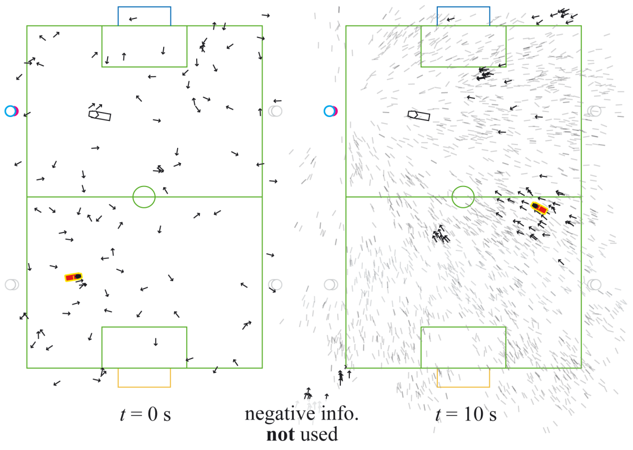


Figure 10: Particle distribution not using negative information, initial uniform distribution and distribution after 10s. Solid arrows indicate Monte Carlo particles (100). The experiment was repeated using 2000 particles (shaded lines) to better represent the actual probability distribution. The actual robot position is indicated by the white symbol, the estimated 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.

entropy, the pose 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 towards the actual robot pose. This is remarkable since without using negative information, localization was not possible.

Using field lines for localization. The previous experiment was repeated using field lines for localization in addition to landmarks. This enables the robot to localize quickly at the actual robot pose even when using the basic localization (fig. 12, right). Adding negative information, however, greatly increases the rate of convergence and the overall level of entropy is reduced even further. The decrease of entropy when incorporating negative information is not obscured by the usage of lines for localization although field lines offer a much greater information content than negative information.

Kidnapped Robot. The kidnapped robot problem is a commonly used benchmark for the flexibility and robustness of localization algorithms [5]: a localized robot is displaced and the time for it to recover is measured. Our kidnapped robot experiments underlined and confirmed the already stated findings. The robot is able to recover from displacements without using negative information as soon as it successively sees three landmarks. In regions where this is not guaranteed, the case is different. Whereas without using negative information, the robot does not have enough evidence to update its

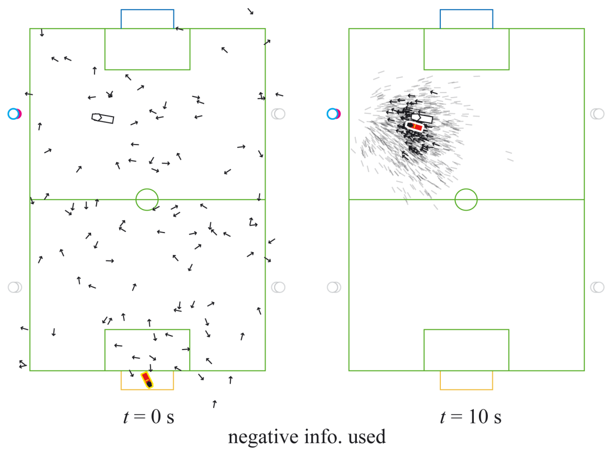


Figure 11: Particle distribution when negative information is incorporated, initial uniform distribution and distribution after 10s. When incorporating negative information, the robot is able to localize quickly.

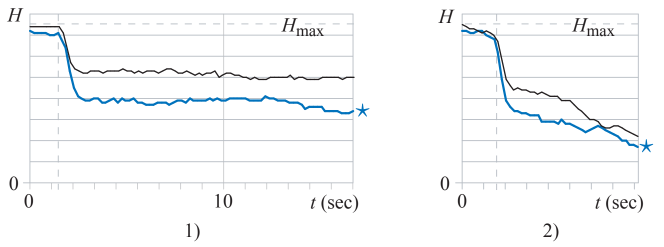


Figure 12: Expected entropy of the belief in the localization task with (⋆) and without (thin line) using negative information. 1) At first the robot does not see the landmark. As soon as the landmark comes into the robot's view (indicated by the dashed vertical line), the entropy drops. Using negative information, the quality of the localization is greatly improved and the entropy continues to decrease over time. 2) Additionally using field lines for localization enables the robot to localize even without negative information. Incorporating negative information, however, yields a higher rate of convergence and the entropy is significantly lowered.

belief, incorporating negative information allows the robot to localize quickly and reliably in such regions.

The ability to localize more quickly using negative information is highly beneficial in real world applications where the robot is trying to actually perform a task rather than to localize perfectly. Such tasks often require the robot to focus its attention on objects other than landmarks and the sensing strategy may keep it from seeing as much of the world as it potentially could. Integrating negative evidence thus allows for more efficient sensing and improves overall robot performance.

5 Conclusion

In this paper we demonstrated how integrating negative information as well as information about collisions into Markov localization can be used to achieve significantly better localization performance for a mobile robot.

An odometry-based motion model is improved using the knowledge about collisions with obstacles yielding a quality measure for the odometry data. This knowledge is 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 is reduced and extra noise is 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. Because sensors are more likely to overlook observable landmarks than hallucinate ones that are not visible, extra care has to be taken in designing the sensor model. 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.

The additional information that is being incorporated into the belief makes it more responsive. This improves localization in areas where there are few landmarks visible and, on the other hand, leads to a quick degradation of the belief when collisions occur. The latter is often the case when two robots fight over a ball and one tries to shot the ball; such action often fails because the robot is unaware of being badly localized and then shoots the ball in an undesirable direction. Incorporating collision detection into the belief allows the robot to recognize such situations and act accordingly.

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. The increased responsiveness of the probability distribution will allow for active vision approaches that take the current belief into account.

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. Original implementation of the localization algorithm by Thomas Röfer. Freek Stulp pointed out “Silver Blaze” to us.

References

[1] J. E. Clark, J. G. Cham, S. A. Bailey, E. M. Froehlich, P. K. Nahata, R. J. Full, and M. R. Cutkosky. Biomimetic Design and Fabrication of a Hexapedal Running Robot. In *Intl. Conf. Robotics and Automation (ICRA2001)*, 2001.

[2] D. Fox, W. Burgard, F. Dellart, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. In *Proc. of AAAI*, 1999.

[3] D. Fox, W. Burgard, and S. Thrun. Active markov localization for mobile robots. In *Robotics and Autonomous Systems*, 1998.

[4] M. Fujita and H. Kitano. Development of an Autonomous Quadruped Robot for Robot Entertainment. *Autonomous Robots*, 5(1):7–18, 1998.

[5] J.-S. Gutmann and D. Fox. An experimental comparison of localization methods continued. *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2002.

[6] J. Hoffmann, M. Jüngel, and M. Lötzsch. A vision based system for goal-directed obstacle avoidance. In D. Nardi, M. Riedmiller, C. Sammut, and J. S.-V. (Eds.), editors, *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, volume 3276 of *Lecture Notes in Artificial Intelligence*, pages 418–425. Springer, 2005.

[7] 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.

[8] W. Koch. Utilizing negative information to track ground vehicles through move-stop-move cycles. In *Proceedings of the SPIE*, volume 5429, pages 273–283, 2004.

[9] 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. to appear.

[10] A. Lankenau, T. Röfer, and B. Krieg-Brückner. Self-Localization in Large-Scale Environments for the Bremen Autonomous Wheelchair. In *Spatial Cognition III*, Lecture Notes in Artificial Intelligence. Springer, 2002.

[11] 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.

[12] S. Lenser and M. Veloso. Visual sonar: Fast obstacle avoidance using monocular vision. In *Proceedings of IROS'03*, 2003.

[13] M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using fastslam. 2003.

[14] C. L. P. Dario, E. Guglielmelli. Humanoids and personal robots: design and experiments. *Journal of Robotic Systems*, 18(2), 2001.

[15] T. Röfer. Evolutionary Gait-Optimization Using a Fitness Function Based on Proprioception. In D. Nardi, M. Riedmiller, C. Sammut, and J. S.-V. (Eds.), editors, *8th International Workshop on RoboCup 2004 (Robot World Cup Soccer Games and Conferences)*, volume 3276 of *Lecture Notes in Artificial Intelligence*, pages 310–322. Springer, 2005.

[16] 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.

[17] 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.

[18] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*, page 231. MIT Press, 2005.

[19] 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.

[20] T. Weigel, A. Kleiner, F. Diesch, M. Dietl, J.-S. Gutmann, B. Nebel, P. Stiegeler, and B. Szerbakowski. Cs freiburg 2001. In *RoboCup 2001 International Symposium*, Lecture Notes in Artificial Intelligence. Springer, 2003.

[21] K. Yoshida, H. Hamano, and T. Watanabe. Slip-Based Traction Control of a Planetary Rover. In *Experimental Robotics VIII*, Advanced Robotics Series. Springer, 2002.