Volume 55, issue 12, 31 December 2007
ISSN 0921-8890
ELSEVIER
Robotics and Autonomous Systems
Special Issue:
Robotics and Autonomous Systems in the 50th Anniversary of Artificial Intelligence
Guest Editors:
Alicia Casals and Antonio Fernández-Caballero
Available online at
ScienceDirect
www.sciencedirect.com

Available online at www.sciencedirect.com

ScienceDirect

Robotics and Autonomous Systems 55 (2007) 870–880

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Localization of legged robots combining a fuzzy-Markov method and a population of extended Kalman filters[☆]

Francisco Martín*, Vicente Matellán, Pablo Barrera, José M. Cañas

Robotics Lab, Rey Juan Carlos University, C/ Tulipán, 28933 Móstoles, Madrid, Spain

Available online 20 September 2007

Abstract

This paper presents a new approach to robot vision-based self-localization in dynamic and noisy environments for legged robots when efficiency is a strong requirement. The major contribution of this paper is the improvement of a Markovian method based on a fuzzy occupancy grid (FMK). Our proposal combines FMK with a population of Extended Kalman Filters, making the complete algorithm both robust and accurate while keeping its computational cost bounded. Two different strategies have been designed to combine both the methods. They have been tested in the RoboCup environment and quantitatively compared with other approaches in several experiments with the real robot.
© 2007 Elsevier B.V. All rights reserved.

Keywords: Mobile robots; Robotic soccer; Localization; Fuzzy logic; Kalman filter; Legged robot

1. Introduction

For 50 years one of the recurring testbeds for artificial intelligence has been the chess. Chess is a highly abstract problem that can be easily represented on a computer. This problem was used to focus the efforts of AI researchers on this complex common problem. In 1997, Deep Blue defeated Garry Kasparov and achieved this milestone for AI. More challenging problems have been proposed to keep AI progressing. One of the most popular is the RoboCup [5]. RoboCup is an attempt to foster AI and intelligent robotics research by providing a standard problem in which a wide range of technologies can be integrated and examined. The ultimate goal of the RoboCup project is to develop a team of fully autonomous humanoid robots that can defeat the human world champion team in soccer by 2050. Many problems have to be solved to achieve this goal: perception, locomotion, collaboration, localization, etc.

In this paper we will focus on the localization problem in legged robots in the RoboCup environment. The self-localization capacity of mobile robots [13] can be defined as the ability to determine its position in the world using its own sensors. Localization problem includes the ability of globally localizing the robot from scratch, or from an erroneous position, (global localization) and the ability of tracking the robot from a known pose (local localization) [18]. The techniques that are used to solve this problem vary enormously depending on the sensors available in each type of robot.

Probabilistic approaches have been successfully implemented to deal with the existing uncertainties in the real world. For example, the work by Simmons [14] showed how Bayesian methods can be used to determine the robot's pose in an office environment. Other probabilistic approaches are those based on Monte Carlo that have been successfully used in single robots [9] or in multi-robot environments [10]. Condensation is a Monte Carlo approach that is used in [18] to localize a vision-based robot in an crowded indoor environment. In this work, the occlusion problem due to the people surrounding the robot is solved using the images of the camera pointing at the ceiling. Monte Carlo techniques have been widely used in the RoboCup environment, Sensor Resetting Localization (SRL) [2] is a good example. Kalman Filtering is also a classic approach [1] to estimate the robot's pose. A comparison of Extended Kalman Filtering and sequential Monte Carlo methods can be found in [12], and a more general comparison can be found in [11].

[☆] This work has been partially funded by the Ministerio de Ciencia y Tecnología, in the ACRACE project: DPI2004-07993-C03-01 and by the Comunidad de Madrid in the RoboCity 2030 project: S-0505/DPI/0176.

* Corresponding author.
E-mail addresses: fmartin@gsyc.es (F. Martín), vmo@gsyc.es (V. Matellán), barrera@gsyc.es (P. Barrera), jmplaza@gsyc.es (J.M. Cañas).

0921-8890/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.robot.2007.09.006

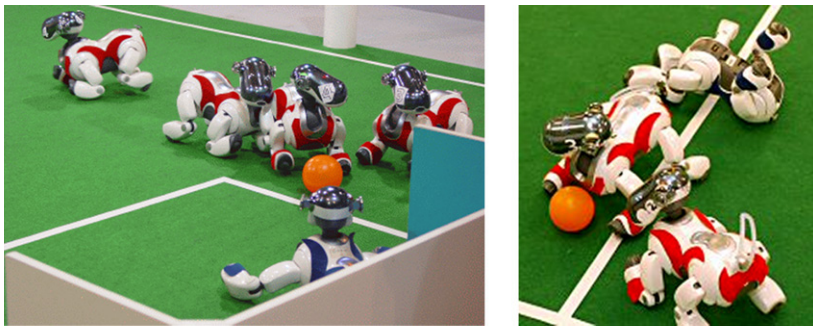


Fig. 1. Problems in the legged robot odometry.

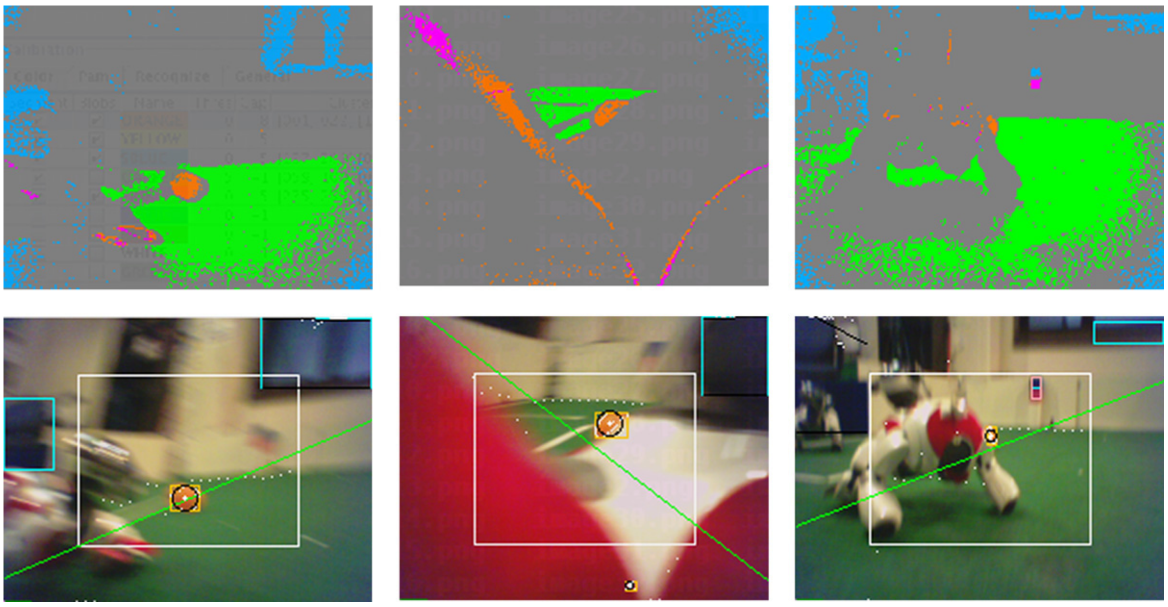


Fig. 2. Color segmentation, and recognized objects superposed on the real image.

Most of these works have proposed solutions to the location problem for wheeled robots using rich 360° range sensors such as sonar or laser, and with accurate odometric information available. But they cannot be applied to legged robots, which only have directional vision and do not have reliable odometry. Our proposal has been designed for that scenario and tested in the AIBO legged robot. Other relevant localization works for this kind of robots are [15,7], both using a Monte Carlo approach, and [8] which uses a Extended Kalman Filter.

The AIBO legged robot is used in the four legged league[1] competition of the RoboCup Federation (Fig. 1). In this competition, robots must be aware of their position on the field at any moment so that they can adapt their behavior. For instance, robots have to be well-located in order to know when to kick, or to generate a group strategy among the members of a team. The robots are not allowed to abandon the field and they also have to be able to place themselves at a predefined starting point when the match is starting and after each goal.

The RoboCup playing field is a 6 × 4 m rectangle. In the center of both small sides are the goals, and on the other sides there are four beacons. All these landmarks are colored so that they can be easily detected by filtering the images obtained by the robots. Other landmarks of the field, such as lines, can also be used for localization in the environment. The light conditions are also controlled. In spite of these facilities, the algorithms developed for localization must consider that both perception and locomotion can be very noisy or erroneous. Robots collide with each other, making the odometry sent to the Localization module highly erroneous, as shown in Fig. 1 (in the left image robot numbered 4 is being pushed, and in the right one, one of the robots has been turned around). Also, it had to be able to recover from situations in which the robot has suddenly been moved to another place of the field (kidnapped problem). In the RoboCup environment, this situation frequently happens when the robot is penalized by the referee and it is manually removed for thirty seconds from the game field.

Perception is also challenging, as shown by Fig. 2. In the leftmost image, the fast robot movements causes failure in the detection of the landmark; in the center image, the robot vision has been partially occluded by other robots, and in the rightmost image one external object (the window) is taken as the blue goal. The upper part of Fig. 2 is the resulting color segmentation made, and the lower part shows that the recognized objects

[1] http://www.tzi.de/4legged/bin/view/Website/WebHome.

Table 1
Localization methods used in the RoboCup 2005

Method	Number of teams
Monte Carlo	12
EKF	3
Monte Carlo + EKF	3
Triangulation	2
Fuzzy based	1

are superposed to the original image (marked as squares for landmarks and goals, and as a circle for the ball).

Several localization methods have been used in the teams competing at RoboCup. According to the self-description of the teams (last edition completely published when writing this paper was the 2005 edition), most of the teams used methods based on Particle Filters (Monte Carlo localization). Six teams used Extended Kalman Filters (EKF), three of them combined with Monte Carlo [12]. Our team, TeamChaos,[2] used a fuzzy logic Markov method (FMK), described in [4,16]. This information has been summarized in Table 1.

CPU requirements of each of the modules that generates the complex behavior are also critical in this scenario. In particular, our method based on fuzzy logic [6] solves the problem of global localization, but consumes a lot of the robot resources and its needs grow exponentially with the accuracy and the size of the field. In addition, this FMK method have shown to be really unstable without very precise tuning, so we decided to study the combination of this method with Extended Kalman Filter (EKF).

In the remainder of this paper we first detail the fuzzy method which we want to improve in Section 2. Next, in Section 3 we describe the Extended Kalman Filter implementation adapted to our problem, and in Section 4, we explain how it combines with the fuzzy FMK method to solve the localization problem, both global and local, in our environment. The results of the experiments are presented in Section 5 and the discussion and future works are explained in Section 6.

2. Fuzzy-logic-based localization method

The initial goal of this method of global localization was to provide the robots with a robust way of representing the uncertainty about its position. Next, we summarized this method, named FMK, because it is the basis of the improvement proposed on this article. The method was initially developed by TeamSweeden [4], and adapted by TeamChaos [6].

In FMK, the field is represented by a grid G_t so that $G_t(x, y)$ is the probability of finding the robot at a position (x, y). Each one of the positions of this grid is a cell of configurable dimension. Each one of the cells contains information of the probability that the robot is in a determined cell, and information on the most probable orientation range. This means

[2] http://www.teamchaos.es.

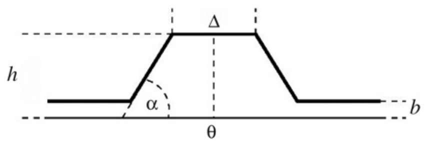


Fig. 3. Diffuse trapezoid.

that it is actually a $2\frac{1}{2}D$ grid because only one orientation is represented.

This information is represented by a diffuse trapezoid (Fig. 3). This trapezoid is defined by the tuple $\langle \theta, \Delta, \alpha, h, b \rangle$. Intuitively, if h is low, the probability of being in this cell is low. If h is high, it is very probable that the robot is in this position. If the trapezoid is wide (Δ is large), great uncertainty exists about the orientation of the robot. If the trapezoid is narrow, or even has triangular form (because Δ is practically null), the orientation uncertainty is so low that we can affirm that the robot orientation is θ.

The localization process used in this method is iterative, each cycle having a prediction and an update step. The prediction step is carried out whenever a movement is performed blurring the probability grid in the direction of movement, using the odometry. In the update step the visual information is included. Our sensor model is based on the information of distance and bearing with respect to the 6 landmarks of the playground.

The robot is considered localized when the quality is high. The quality depends on the size of the field with cells with high probability. If the cells with high probability (their component h is high) are concentrated in a reduced area of the field, the quality is higher than that in the case of the cells with high probability covering big areas of the field, or when representing several hypothesis (cells with high probability are concentrated in several areas). The robot position is determined in the center of the area with high probability cells.

An example of the sequential application of the prediction and update steps can be observed in Fig. 4. The white color indicates cells with high probability, and the black color indicates cells with low probability. The red circle is the robot's position, and the red arrow is the orientation. A robot is considered localized when there are few cells with high probability (colored white) in an area. In the first figure (the leftmost figure), the robot starts from a situation of total ignorance because all the cells have the same probability and Δ value of each cell covers all the orientations. Using the sensorial information relative to the distance to the goal (center figure) the robot is able to improve its position belief, having higher probability the cells are set at the perceived distance from goal. When the robot moves, the grid is blurred because of the unreliable odometry, making the zone with high probability wider.

The performance of the entire method depends almost exclusively on the size of each cell of the grid. If the size of the cell is large, the accuracy is low, and the computational cost is also low. If more accuracy is required, the cell size has to be reduced, but the computational cost can make the method unusable because it is increased almost exponentially with the size of the grid. In practice, a 20 × 20 cm cell was adequate to

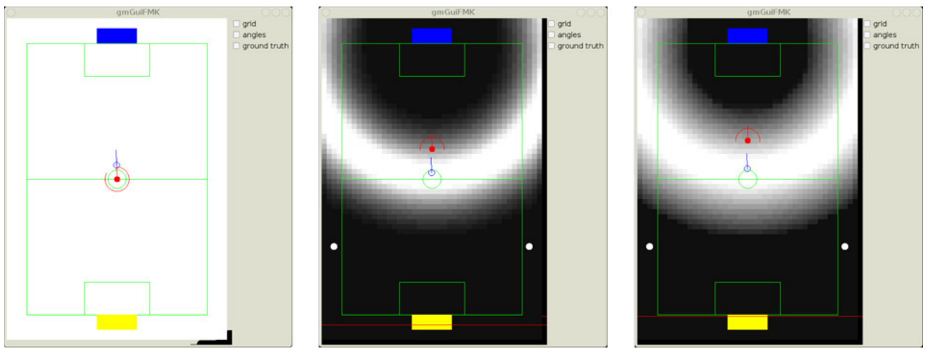


Fig. 4. Process of localization of the robot in the fuzzy grid. Probability is represented by greyscale, being white cells the most probable ones, and black cells the less probable ones. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

obtain a reasonable level of accuracy in the small field previous to the 2005 rules change.

This localization method has shown the following properties:

- Fast recovery from erroneous or unknown estimations.
- Fast recovery from kidnappings.
- Multi-hypothesis in the (x, y) estimation.
- Much faster than classical Markovian approaches, in which the orientation is composed of several (x, y) states. In this method, the orientation estimation is a simple trapezoid operation.

The drawbacks, that motivates this work, are the following:

- This method is difficult to tune.
- Mono-hypothesis in the *theta* estimation.
- Very sensitive to sensor errors and false positives, which makes the method very unstable in noisy conditions.
- The computation time became unacceptable when the field size was sensibly incremented in 2005 (356 $\rightarrow$ 600 cells). The time used by the localization module became very high if the previous accuracy was kept. This is the main reason for developing a technique based on extended Kalman filters.

3. Localization method using the extended Kalman filter

The EKF is one the most popular tools for state estimation in robotics. It is a local localization method whose strength lies in its simplicity and its computational complexity $O(k^{2.4} + n^2)$, where k is the dimension of the measure vector and n the dimension of the state vector [3]. In our implementation, the computational complexity is $O(2^{2.4} + 3^2)$. Its flawlessness lies in its sensibility to noisy measures. In these conditions, the filter diverges and it is difficult to recover from these situations.

The filter is initialized in a known position with given uncertainty about its pose. If this information is not available because of a total ignorance about the robot pose, the filter is initialized with the robot pose at the center of the field and a high enough uncertainty to consider the robot in any position in the field.

In order to implement an EKF we have defined the position of a robot as the state vector $s \in \Re^3$,

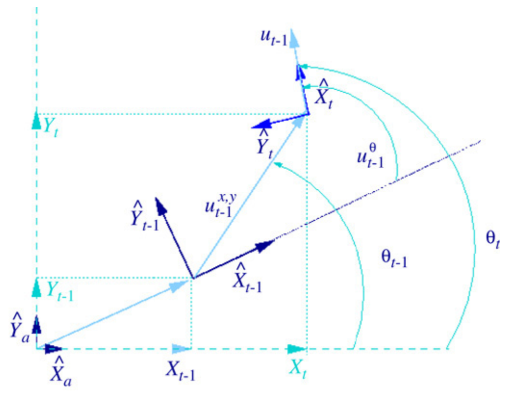


Fig. 5. Odometry-based movement model.

$$\mathbf{s} = \begin{pmatrix} x_{\text{robot}} & y_{\text{robot}} & \theta_{\text{robot}} \end{pmatrix}^{\mathrm{T}}. \tag{1}$$

The update process will be guided by two nonlinear functions, f and h. First, f relates the previous state s_{t-1}, the odometry u_{t-1}, and the noise in process w_{t-1}, to the present state s_t, according to Fig. 5.

$$s_t = f(s_{t-1}, u_{t-1}, w_{t-1}). \tag{2}$$

3.1. Prediction step

In this step $s_t^- \in \Re^3$ is calculated. This is the position in which the robot will be, according to its previous position and the information of the odometry using the movement model of Fig. 5. Also $P_t^- \in \Re^{3\times 3}$ is calculated. This matrix represents the general error of the system. It is initialized with the maximum possible error: the field dimensions for x, y, and 180 for θ.

$$s_t^- = \begin{pmatrix} x_t^- & y_t^- & \theta_t^- \end{pmatrix}^{\mathrm{T}} = f(s_{t-1}, u_{t-1}, 0) = \begin{pmatrix} x_{t-1} \\ y_{t-1} \\ \theta_{t-1} \end{pmatrix} + \begin{pmatrix} (u^x_{t-1} + w^x_{t-1})\cos\theta_{t-1} - (u^y_{t-1} + w^y_{t-1})\sin\theta_{t-1} \\ (u^x_{t-1} + w^x_{t-1})\sin\theta_{t-1} + (u^y_{t-1} + w^y_{t-1})\cos\theta_{t-1} \\ u^{\theta}_{t-1} + w^{\theta}_{t-1} \end{pmatrix} \tag{3}$$

$$P_t^- = A_t P_{t-1} A_t^{\mathrm{T}} + W_t Q_{t-1} W_t^{\mathrm{T}}. \tag{4}$$

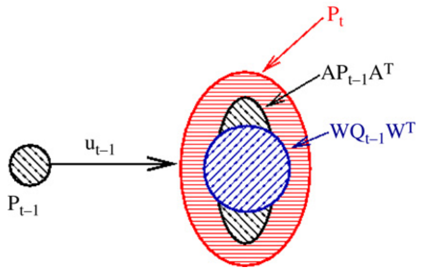


Fig. 6. Evolution from P_{t-1} to P_t^-.

$A_tP_{t-1}A_t^{\mathrm{T}}$ represents the previous uncertainty P after control input u (Fig. 6). A_t is defined as follows:

$$A_t = \frac{\partial f}{\partial s} = \begin{pmatrix} 1 & 0 & -u_{t-1}^{y}\cos\theta_{t-1} - u_{t-1}^{x}\sin\theta_{t-1} \\ 0 & 1 & u_{t-1}^{x}\cos\theta_{t-1} - u_{t-1}^{y}\sin\theta_{t-1} \\ 0 & 0 & 1 \end{pmatrix} \quad (5)$$

Q_t represents the noise in the prediction process. $W_tQ_{t-1}W_t^{\mathrm{T}}$ represents the noise Q added to P_t^- from P_{t-1}. We have experimentally determined that the information given by the odometry system can be represented by a normal distribution $N(0, 0.3u_{t-1})$. Applying transformation matrix W_t (Eq. (7)), this noise is introduced in the uncertainty.

$$Q_t = E[w_t w_t^{\mathrm{T}}]$$
$$= \begin{pmatrix} (0.3u_{t-1}^{x})^2 & 0 & 0 \\ 0 & (0.3u_{t-1}^{y})^2 & 0 \\ 0 & 0 & (0.3u_{t-1}^{\theta} + \frac{\sqrt{(u_{t-1}^{x})^2 + (u_{t-1}^{y})^2}}{500})^2 \end{pmatrix} \quad (6)$$

$$W_t = \frac{\partial f}{\partial w} = \begin{pmatrix} \cos\theta_{t-1} & -\sin\theta_{t-1} & 0 \\ \sin\theta_{t-1} & \cos\theta_{t-1} & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (7)$$

In a typical implementation of an EKF, this step is followed by the update step to get the new s_t and P_t. In this work, the frequency of the odometry readings is higher than the sensor measures. For this reason, the prediction and correction step are executed independently making $s_t = s_t^-$ and $P_t = P_t^-$ at the end of each step.

3.2. Update step

Our sensor model is based on the information about the 6 landmarks of the playground $m_{1\ldots6}$, each one is placed in a known position (x, y). Information z_t^i perceived from landmark i at time t is the vector $(r_t^i, \phi t^i)$, representing the distance and the orientation to that landmark. For each perception cycle, a measure i updates the system state as:

$$s_t = s_{t-1} + K_t^i(z_t^i - \hat{z}_t^i) = s_{t-1} + K_t^i(z_t^i - h^i(s_{t-1})) \quad (8)$$

where $h_i(s_{t-1})$ is a nonlinear function (Eq. (9)) that calculates the $\hat{z}_t^i$ predicted depending on the robot pose estimation,

$$\hat{z}_t^i = h^i(s_{t-1}) = \begin{pmatrix} \text{distance}(m^i, s_{t-1}) \\ \text{angle}(m^i, s_{t-1}) \end{pmatrix}$$
$$= \begin{pmatrix} \sqrt{(m_{t,x}^i - s_{t-1,x})^2 + (m_{t,y}^i - s_{t-1,y})} \\ \text{atan2}(m_{t,x}^i - s_{t-1,x}, m_{t,y}^i - s_{t-1,y}) - s_{t-1,\theta} \end{pmatrix} \quad (9)$$

K_t^i, known as the *Kalman gain*, is calculated as:

$$K_t^i = P_{t-1}(H_t^i)^{\mathrm{T}}(S_t^i)^{-1} \quad (10)$$

$$S_t^i = H_t^i P_{t-1}(H_t^i)^{\mathrm{T}} + R_t^i \quad (11)$$

$$H_t^i = \frac{\partial h^i(s_{t-1})}{\partial s_t}$$
$$= \begin{pmatrix} -\frac{m_{t,x}^i - s_{t-1,x}}{\sqrt{q}} & -\frac{m_{t,y}^i - s_{t-1,y}}{\sqrt{q}} & 0 \\ \frac{m_{t,y}^i - s_{t-1,y}}{q} & -\frac{m_{t,x}^i - s_{t-1,x}}{q} & -1 \\ 0 & 0 & 0 \end{pmatrix} \quad (12)$$

$$q = (m_{t,x}^i - s_{t-1,x})^2 + (m_{t,y}^i - s_{t-1,y})^2$$

R_t^i represents the noise in the sensor information. Depending on the type of landmark (net or beacon) the estimated errors are shown in Fig. 7. This observation model was empirically made, and reflects how the distance estimation error is big when the robot is far from the landmark, and how the angle estimation error is low when the robot perceives the entire landmark. The resulting covariance error in the robot position estimation P_t is calculated as $(I - K_t^i H_t^i)P_{t-1}$.

3.3. Outliers filter

Before incorporating all the z_t^i sensed, a δ_t^i value is calculated (Eq. (13)). This value represents the measurement likelihood and it is used to reject those measurements considered false positives or unknown correspondences. If this value is higher than a threshold, the measure is rejected and it is not incorporated into the filter. This threshold changes dynamically in the range heuristically set between 5 and 100.

$$\delta_t^i = (z_t^i - \hat{z}_t^i)^{\mathrm{T}}(S_t^i)^{-1}(z_t^i - \hat{z}_t^i). \quad (13)$$

4. Combining fuzzy logic and extended Kalman filter

On the one hand, FMK offers a global method able to recover quickly from situations of complete uncertainty. However, this method consumes many computational resources, even using the compact way of representing the angular information of each cell. A possible solution so that the method was usable would have been to increase the size of each cell of the grid, but this means less accuracy in the estimation of the position of the robot. This method also fails in situations with erroneous observations. In this case, the system estimates the robot position in the center of the filed, as in the initial step with full uncertainty.

On the other hand, the local method of localization based on EKF is computationally light and very accurate. The problem of this method is that the robot is not able to self-locate quickly when starting from a situation of total uncertainty. Neither, is it able to recover from situations of high error in the estimations, nor from the manual change of the position of the robot during the game.

Combining both local and global methods we hope to obtain several advantages. FMK estimation will help to initialize the

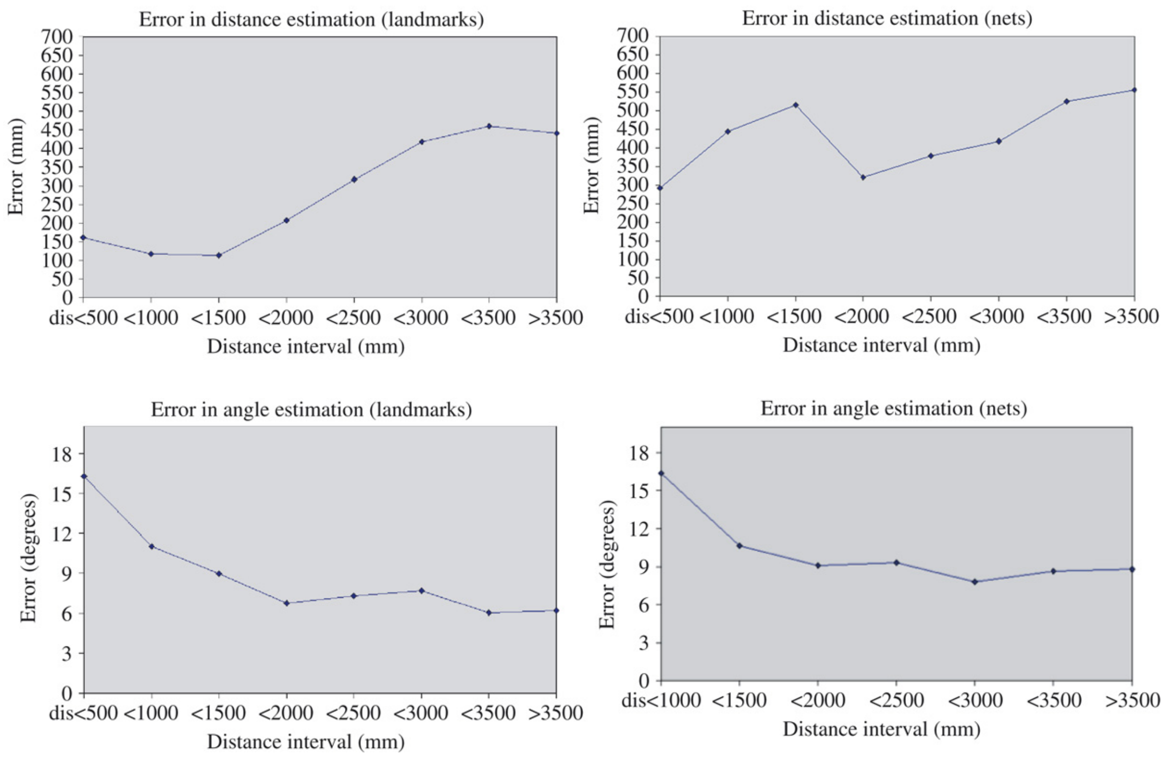


Fig. 7. Errors in the distance and angle estimation depending on the type of landmark and the actual distance to it.

EKF filter in the case of total ignorance. The FMK grid helps to evaluate if the EKF filter may be wrong. Once initialized, the EKF filter is more stable than FMK when erroneous observations are detected. If the cell size in FMK is increased, obtaining a better processing time with loss of accuracy, the EKF filter provides the accuracy needed.

Two different strategies are proposed in this work to combine both methods. In the first approach, an EKF runs concurrently with the FMK. In the second approach, several EKFs run concurrently with the FMK. In both the strategies the cell size for FMK has been increased to 500 mm. This allows us to obtain a fast global localization with low accuracy, That is compensated by accuracy obtained from the EKF method.

We will describe both approaches in detail in the next subsections, and then the experiments to evaluate them.

4.1. FMK + EKF

This proposal combines FMK with one EKF. The combination in this case is simple, as shown in Algorithm 1. Both algorithms are initialized to full uncertainty: all the FMK cells are identically probable, so the position calculated is the center of the field; EKF position is set at the center of the field, with an uncertainty that covers all the field.

At the end of each cycle, the position estimated by EKF is compared with the FMK information. To make this comparison, the value of the cell on the position estimated by EKF is used. If this value is low, the EKF estimation is supposed to be erroneous. If FMK quality is low, it makes no sense to reset. In the same way, If the position estimated by FMK is close enough (only one cell apart) to the EKF estimation, we let the filter converge itself.

In order to summarize, we only reset the EKF filter when we consider that the FMK estimation is reliable, far from the EKF estimation, which is improbable according to the FMK information.

Algorithm 1: $FMK + EKF$ combination strategy.

```
Initialize $pos_{fmk}$ using full uncertainly;
Initialize $pos_{ekf}$ using full uncertainly;
while true do
    Predict $pos_{fmk}$ using odometry;
    Predict $pos_{ekf}$ using odometry;
    Correct $pos_{fmk}$ using landmark information;
    Correct $pos_{ekf}$ using landmark information;
    if $Prob(cell_{fmk}(pos_{ekf,x}, pos_{ekf,y})) < threshold$ then
        if (quality($pos_{fmk}$) is high) and (distance($pos_{fmk}$, $pos_{ekf}$) is large) then
            Initilize $pos_{ekf}$ to $pos_{fmk}$
        end
    end
    robot position $\leftarrow pos_{ekf}$;
end
```

4.2. FMK + (EKF × n)

Using the previous strategy, incorrect estimations appear when the FMK estimation has low uncertainty, but it is erroneous. It is not very common, but it is difficult to recover quickly from this situation, as preliminary experiments have shown. For this reason, we have also tried to maintain several hypotheses, each one represented by an independent EKF. We will still use FMK to give the population of EKFs the global localization ability.

Each EKF has its own estimation about the robot pose, and its own associated uncertainty. They are independent of each other.

The number of EKF filters is not constant. It can be dynamically modified up to a maximum limit. Initially, there will not be any active EKF. Every time that FMK seems reliable, but no filter close enough to the position estimated by FMK, a new EKF filter is created. The new EKF filter is initialized to the center of the FMK cell position, and the uncertainty is set to the one associated with that cell.

EKF filters can also be removed in two situations:

- When the FMK information is reliable and indicates that the position estimated by EKF is not the cell estimated by FMK.
- If two EKFs are very close, there is no sense in maintaining both. The one with less uncertainty is maintained, the other one is removed.

The estimation of the robot position is determined by the EKF filter with less uncertainty. If an EKF is initialized in a wrong position, it is not likely to reduce its uncertainty with the subsequent observations. This filter will not be taken into account to determine the robot position. A filter that estimates the right position, will get its uncertainty reduced at each observation, maintaining its uncertainty low. In the event of a kidnapping, the filter created in the new position will get its uncertainty reduced, and the filter in the old position will get its own increased. In a few steps, the uncertainty of the new filter will be lower than the old one, thus estimating the robot's position correctly. In Algorithm 2 the entire process is detailed.

Algorithm 2: FMK with multiple EKF combination strategy.

```
Initialize $pos_{fmk}$ using full uncertainly;
while $true$ do
    Predict $pos_{fmk}$ using using odometry;
    foreach $active\ ekf_i$ do
        Predict $pos_{ekf_i}$ using odometry;
    end
    Correct $pos_{fmk}$ using landmark information;
    foreach $active\ ekf_i$ do
        Correct $pos_{ekf_i}$ using landmark information;
    end
    Correct $pos_{ekf}$ using landmark information;
    foreach $active\ ekf_i$ do
        foreach $active\ ekf_j$ do
            if $distance(pos_{ekf_i}, pos_{ekf_j}) < threshold_{near}$ then
                if $uncertainly(ekf_i) > uncertainly(ekf_j)$ then
                    remove $ekf_i$;
                else
                    remove $ekf_j$;
                end
            end
        end
        if $Prob(cell_{fmk}(pos_{ekf,x}, pos_{ekf,y})) < threshold$ then
            if $(quality(pos_{fmk})$ is high) and (distance($pos_{fmk}, pos_{ekf}$) is large) then
                remove $ekf_i$;
            end
        end
    end
    if $(quality(pos_{fmk})$ is high) and (distance($pos_{ekf_j}, pos_{ekf_i}$) is large) $\forall i, j$ then
        Initialize $ekf_i \leftarrow fmk$;
    end
    if there is any active filter then
        robot position $\leftarrow pos_{ekf_i}$ where $i$ is the filter with lower uncertainly;
    else
        robot position $\leftarrow pos_{fmk}$;
    end
end
```

5. Experiments

The environment in which the experiments are carried out is the field used in RoboCup. We have used two cenital cameras to track the robot during the experiments to obtain the ground truth (Fig. 8), using a modified version of Mezzanine [17] software to support the two cameras. The robot's behavior used in the experiment tries to patrol a set of predefined way-points. In the experiments, the 4 localization methods described in this article are compared among themselves and with the Sensor Resetting Localization (SRL) method [2]:

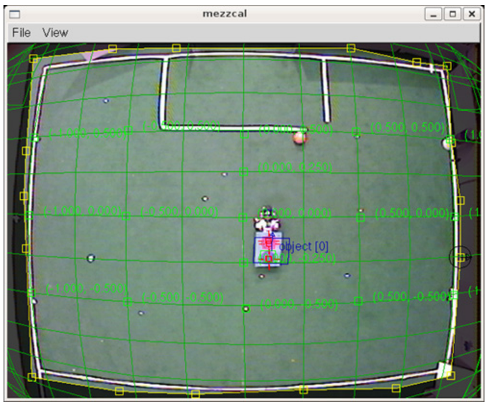


Fig. 8. Tracking system used in the experiments.

- FMK.
- EKF.
- FMK + EKF.
- FMK + (EKF × n).
- SRL (Sensor Resetting Localization).

In these experiments we will measure several aspects: the accuracy in the position and the orientation estimation, the recovery time from an unknown position, the amount of time the robot is successfully localized and the CPU time consumed by each method. The experiments measure how well the robot is localized while performing simple and complex movements, and in the case of kidnapping. Each experiment has been repeated several times to obtain reliable results. In each experiment the conclusions will be based on the evolution of the error in position and orientation during the experiment, and the statistical analysis of these errors. The main statistical indicators are mean and standard deviation of the error, but the analysis of the median will give us an idea of the amount of time each method maintains the error low.

5.1. Experiment 1

The first experiment (straight in Fig. 9) is a simple movement. The robot starts at one net, and goes to the other one. This experiment was repeated 20 times, and the evolution of the error in position and orientation in one execution is shown in Fig. 10. The error in the position estimation of the 20 executions has been summarized in Table 2, and the error in the orientation estimation is summarized in Table 3.

In this trajectory, EKF is not able to converge from its initial position (center of the field). FMK is able to converge quickly to the right position, but it is unstable (the estimation frequently jumps from one position to another when errors in perception occurs). If we take a look at the orientation graph, FMK is not

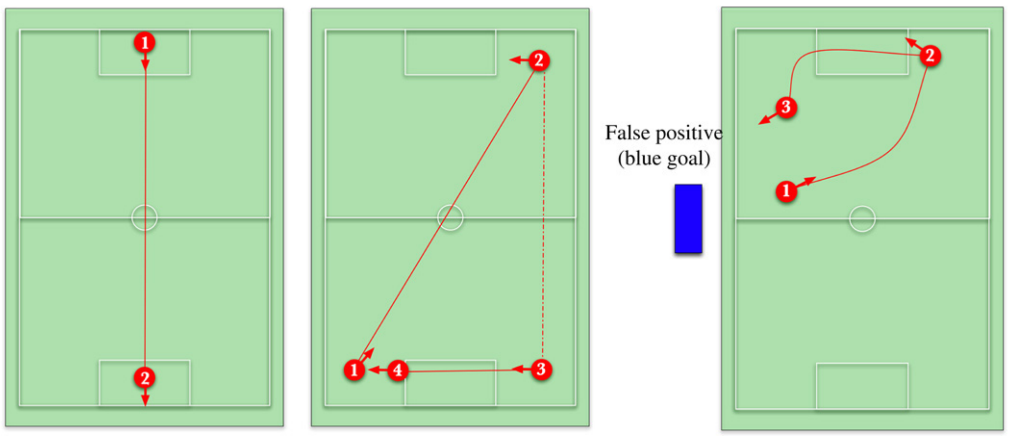


Fig. 9. Path followed in the experiments. The robot follows the path indicated by lines. Each circle is numbered to indicate the initial and ending point.

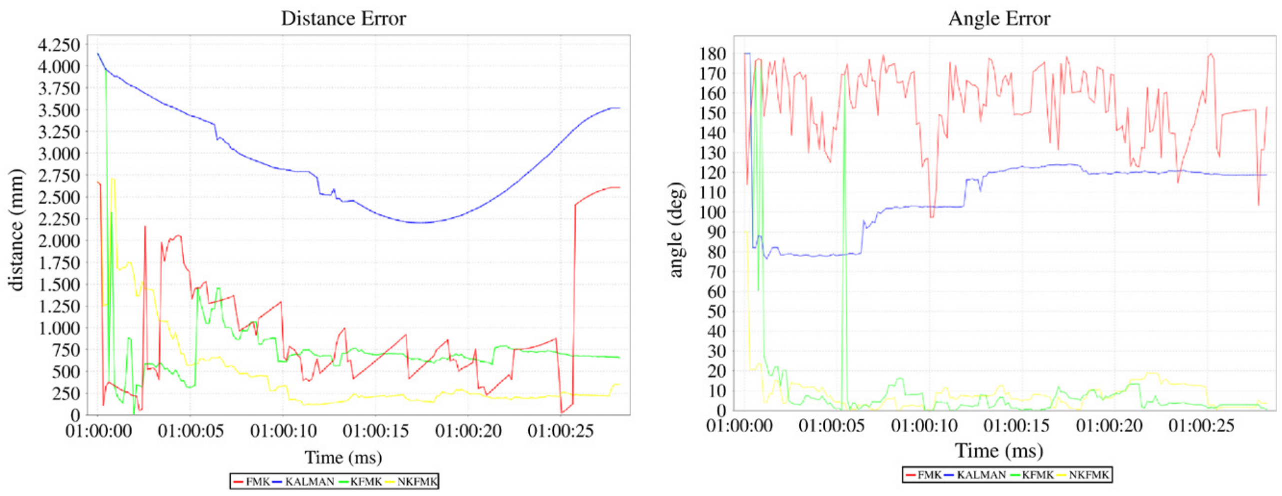


Fig. 10. Error in the distance and angle estimation in one sample trial in the first experiment.

Table 2
Comparison of error in distance (in mm)

Method	Median	Mean	SD
FMK	1626	1668	1041
EKF	2039	1846	968.3
EKF + FMK	1932	1883	1093
n(EKF) + FMK	791.8	1283	1336
SRL	1060	1267	792.3

Experiment 1.

Table 3
Comparison of error in orientation (in deg)

Method	Median	Mean	SD
FMK	91.95	90.49	57.5
EKF	33.05	46.79	38.73
EKF + FMK	81.45	42.71	49.31
n(EKF) + FMK	14.23	32.64	42.97
SRL	24.39	34.05	29.72

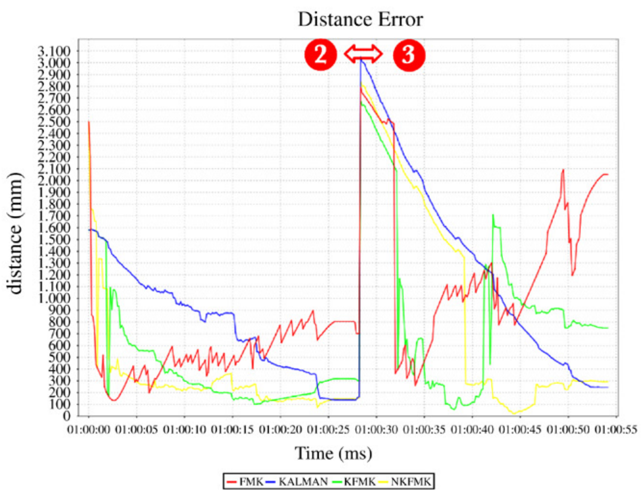
Experiment 1.

able to estimate the angle correctly. The combination of FMK and one EKF converges as quickly as FMK, but it is more stable and robust. Finally, the best accuracy is got by the combination of FMK and several EKFs. The estimation converges due to the FMK information, and is able to be very accurate once it has converged.

5.2. *Experiment 2*

The second experimental tests show how the algorithms manage a kidnapping situation. This experiment was repeated 10 times, and the results are summarized in Table 4, for error in position, and in Table 5 for error in orientation. On the middle of Fig. 9 the path that the robot has followed is represented. It starts from point one, goes to point two, and then it is manually displaced to point three. This situation is usual in the RoboCup matches when the referees penalizes the robot and it returns to the field after two minutes. To see the results more clearly, the instant of kidnapping (2) and release (3) of the robot have been marked in Fig. 11.

The results are similar to the previous experiment until the kidnapping: EKF converges, but very slowly; and the proposed combination of several EKF's is the more accurate. After the kidnapping, single EKF is the most damaged. FMK recovers quickly, although the orientation estimation is very inaccurate. It can also be observed that after the kidnapping, it is unstable,



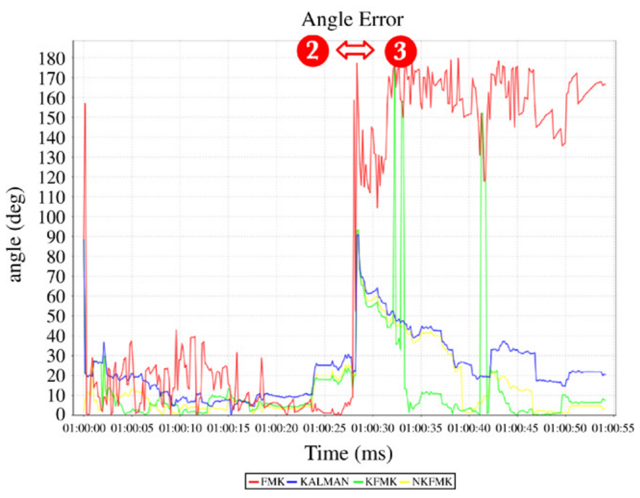


Fig. 11. Error in the distance and angle estimation in the second experiment.

Table 4
Comparison of error in distance (in mm)

Method	Median	Mean	SD
FMK	1649	2060	1551
EKF	1553	1748	838.7
EKF + FMK	1457	1949	1746
n(EKF) + FMK	1361	1761	1331
SRL	1502	1869	1113

Experiment 2.

Table 5
Comparison of error in orientation (in deg)

Method	Median	Mean	SD
FMK	60	77.29	60.22
EKF	18.18	39.21	45.85
EKF + FMK	35.84	55.9	57.22
n(EKF) + FMK	23.46	38.95	38.9
SRL	49.08	66.41	52.31

Experiment 2.

mainly because of the residual cell probabilities before the kidnapping. This makes the combination of FMK and one EKF fail. It is due to its dependability on FMK. The combination of FMK and several EKFs recovers in a reasonable time, and remains stable after the kidnapping.

This experiment also shows some of the drawbacks of the FMK + (EKF $\times$ n) combination strategy. The stability offered by this method makes it slow when recovering from kidnapping. The active filters before kidnapping have good quality and low uncertainty. So, one of them is selected as robot position estimation. When these new erroneous filters start to reject the observation (due the outliers filter) and to incorporate the odometry, the uncertainty grows and the quality will become lower. New filters started after kidnapping in the right position will get their uncertainty reduced, and after several cycles, their uncertainty will become lower than the erroneous ones, recovering from the kidnapping. Sometimes it takes several cycles to incorporate observations and reduce their uncertainty, as shown in Fig. 11.

Table 6
Comparison of error in distance (in mm)

Method	Median	Mean	SD
FMK	610.6	955.6	868.9
EKF	1118	1257	691.3
EKF + FMK	601	946.6	928.8
n(EKF) + FMK	778.3	1226	1227
SRL	1303	1281	602.3

Experiment 3.

Table 7
Comparison of error in orientation (in deg)

Method	Median	Mean	SD
FMK	50.32	70.51	60.43
EKF	9.455	31.61	49.42
EKF + FMK	19.74	38.78	42.6
n(EKF) + FMK	21.05	36.75	39.49
SRL	37.22	45.04	38.55

Experiment 3.

5.3. *Experiment 3*

We designed the last experiment to test the performance of the localization methods when facing a noisy environment and how the EKF outliers filter rejects false positives. In this way, calibration made was not good enough to avoid a couple of false positives. One of them was situated close to the center of one side (blue box in right in Fig. 9) and resulted in the robot perceiving the blue goal in that position. This experiment was repeated 5 times. The error in position is summarized in Table 6, and the error in orientation in Table 7. The data shown in Fig. 12 demonstrates that the combination of FMK and several EKFs is the only algorithm able to estimate the robot position correctly most of the time. The moment when the robot perceives the false positive is marked in both graphs.

5.4. *CPU time analysis*

Besides the accuracy and the convergence speed, the performance is a critical issue in the RoboCup environment.

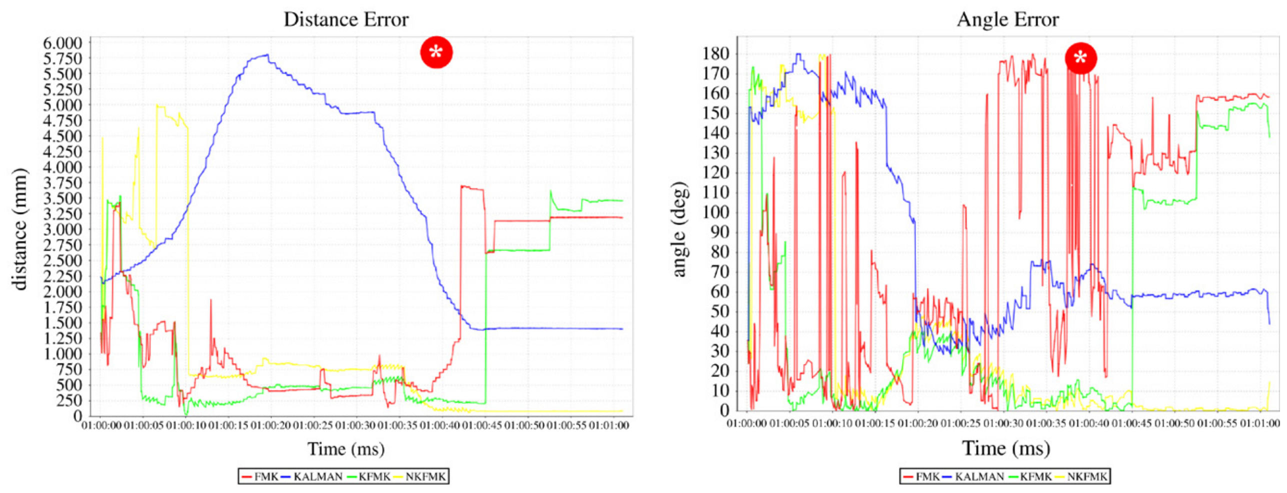


Fig. 12. Error in the distance and angle estimation in the third experiment.

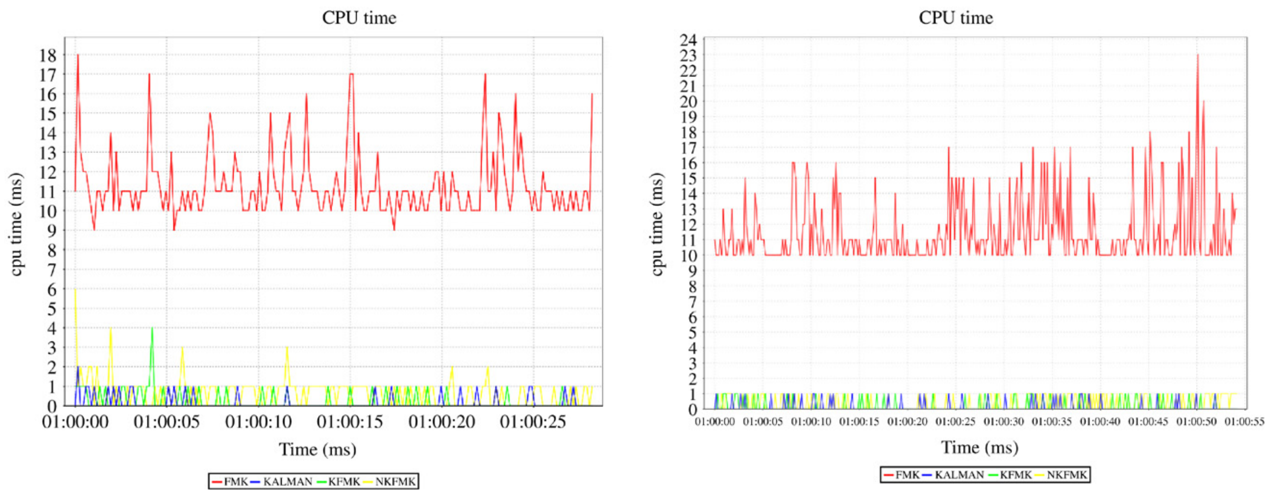


Fig. 13. CPU time spent in every algorithm in experiments 1 and 2.

This has been one of our main concerns. In Fig. 13 the CPU time spent on each algorithm is shown. FMK original (with cell size 20 × 20 cm.) requires up to ten times more CPU than the other methods (which use a FMK version with cell size 50 × 50 cm) almost all the time. This causes the robot to fail, for instance, when it approaches the ball and tries to kick it, because the delay introduced by the localization module makes the robot to kick in an old position of the ball. The new approach has shown an evident upgrade in the robot's performance in kicking and grabbing the ball.

6. Conclusions and future work

The localization methods presented in this article combine the global localization and recovery abilities provided by the Markovian fuzzy logic algorithm with an Extended Kalman Filter, whose advantages are its low processing time requirements and its stability. We have combined them in two strategies. First, in the FMK + EKF strategy the EKF is restarted with the FMK information when the EKF estimated uncertainty is high. Second, in the FMK + (EKF × n) strategy a slightly more complex mechanism has been proposed, running several EKF's in parallel and using the FMK information in some cases to restart the EKF filters.

To validate and compare our methods to others, we have implemented several localization algorithms in a legged Aibo robot and tested them in the RoboCup scenario. This scenario is challenging for localization, as it is an example of dynamic and noisy environment with poor odometry information and directional vision as the main sensor sources.

The FMK + (EKF × n) algorithm has proved adequate for the RoboCup environment constraints and the functional needs. It converges from scratch and recovers from kidnapping. It is also the most robust in noisy environments, maintaining enough accuracy for the robot to perform its tasks with a high level of performance in CPU time. FMK method is able to quickly converge from total uncertain situations and to recover from kidnapping or erroneous estimation situations, but it is very unstable when perception errors occurs. EKF method works fine, but it requires to be properly initialized and it does not recover from kidnapping. The FMK + EKF algorithm let us to initialize the EKF and recovers quickly from kidnapping, but it is not as stable as needed. The FMK + (EKF × n) strategy seems to be more stable than the FMK + EKF. Nevertheless, when kidnapping occurs, the recovery is slower. Regarding computing cost, experiments show that the processing time of the localization module has been greatly reduced compared to the equivalent FMK.

The future works aim to achieve an optimal combination of both algorithms. Also the development of other methods, such as Particle Filters, to reinitiate the Extended Kalman Filter in certain situations, could be useful and should be considered.

Acknowledgments

We want to express our gratitude to the work developed by the TeamChaos four-legged team, and specially to Dr. Humberto Martínez Barberá for his work in the development of tools used in this work, and also for his valuable advice. We would like also to thank Renato Samperio and Prof. Huosheng Hu for their valuable advice and resources provided at the Essex University.

References

[1] Rudolph Emil Kalman, A new approach to linear filtering and prediction problems, Transactions of the ASME–Journal of Basic Engineering 82 (1960) 35–45.

[2] Scott Lenser, Manuela Veloso, Sensor resetting localization for poorly modelled mobile robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, 2000.

[3] Sebastian Thrun, Wolfram Burgard, Dieter Fox, Probabilistic Robotics, MIT Press, ISBN: 0262201623, 2005.

[4] Per Buschka, Alessandro Saffiotti, Zbigniew Wasik, Fuzzy landmark-based localization for a legged robot, in: Proceedings of the International Conference on Intelligent Robots and Systems 2000, vol. 2, Takamatsu, Japan, 2000, pp. 1205–1210.

[5] Hiroaki Kitano (Ed.), RoboCup-97: Robot Soccer World Cup I, in: Lecture Notes in Computer Science, vol. 1395, Springer, 1998.

[6] David Herrero-Pérez, Humberto Martínez-Barberá, Alessandro Saffiotti, Fuzzy self-localization using natural features in the four-legged league, in: Lecture Notes in Computer Science, vol. 3276, Robocup 2004, 2005, pp. 110–121.

[7] Pablo Guerrero, Javier Ruz del Solar, Auto-localización de un robot móvil Aibo mediante el Método de Monte Carlo, Anales del Instituto de Ingenieros de Chile 115 (3) (2003) 91–102.

[8] Raul Lastra, Paul Vallejos, Javier Ruiz-del Solar, Self-localization and ball tracking for the robocup 4-legged league, in: Proceeding of the 2nd IEEE Latin American Robotics Symposium LARS 2005, Sao Luis, Brazil, 2005.

[9] Jürgen Wolf, Wolfram Burgard, Hans Burkhardt, Robust vision-based localization by combining an image retrieval system with Monte Carlo localization, IEEE Transactions on Robotics 21 (2) (2005) 208–216.

[10] Dieter Fox, Wolfram Burgard, Hannes Kruppa, Sebastian Thrun, A Monte Carlo algorithm for multi-robot localization, Technical Report CMU-CS-99-120, Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, 1999.

[11] John Gutmann, Wolfram Burgard, Dieter Fox, Kurt Konolige, An experimental comparison of localization methods, in: Proceedings., 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 2. 1998, pp. 736–743.

[12] David C.K. Yuen, Bruce A. MacDonald, A comparison between extended Kalman filtering and sequential Monte Carlo technique for simultaneous localisation and map-building, in: Proc. Australian Conference on Robotics and Automation, Auckland, New Zealand, 2002.

[13] Johann Borenstein, H.R. Everett, Liqiang Feng, Navigating mobile robots: Systems and techniques, Ltd. Wesley, MA, 1996.

[14] Reid Simmons, Sven Koening, Probabilistic navigation in partially observable environments, in: Proceedings of the 1995 International Joint Conference on Artificial Intelligence, Montreal, Canada, 1995, pp. 1080–1087.

[15] Mohan Sridharan, Gregory Kuhlmann, Peter Stone, Practical vision-based Monte Carlo localization on a legged robot, in: IEEE International Conference on Robotics and Automation, 2005, pp. 3366–3371.

[16] Humberto Martínez, Vicente Matellán, Miguel Cazorla, Teamchaos Technical Report, Technical Report, TeamChaos, 2006.

[17] Andrew Howard, Mezzanine User Manual, Institute for Robotics and Intelligent Systems, Technical Report IRIS-02-416, 2002.

[18] Frank Dellaert, Dieter Fox, Wolfram Burgard, Sebastian 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.

Francisco Martín received his B.Eng. in Computer Science from Rey Juan Carlos University, Spain, in 2003. He is currently working toward the Ph.D. degree at Robotics Group, Rey Juan Carlos University, where he is Lecturer. His research interest include mobile robotics, localization methods and computer vision.

Vicente Matellán got his Ph.D. at the Technical University of Madrid, and worked as Assistant Professor at Carlos III University, currently is an Associate Professor in Computer Science at the Rey Juan Carlos University, leading the Robotics Group (Móstoles-Madrid, Spain). His main research interest include multi-robot systems, robotic software architectures, artificial vision, free software, and distributed systems. He has published over 100 papers in journals, books, and conferences in these areas.

Pablo Barrera. Ph.D. candidate. He graduated in Telecommunication Engineering from Universidad Carlos III de Madrid in 2002. He worked in Universidad Carlos III until 2003. Currently he is working in Universidad Rey Juan Carlos as teaching assistant. His research interests include computer vision and robotics.

José M. Cañas received the Telecommunication Engineer degree in 1995, and a Ph.D. in Computer Science in 2003, both from the Universidad Politéanica de Madrid. He has researched in Carnegie Mellon University and the Instituto de Automática Industrical. Currently he is associate professor at Universidad Rey Juan Carlos, where he coleads the robotics group. His research interests include robotics and artificial vision.