Extended Kalman filter populations for a reliable real-time robot self-localization

Francisco Martín, Carlos Agüero and José María Cañas
Robotics Lab, Universidad Rey Juan Carlos

***Abstract*— Autonomous mobile robots need a reliable and accurate self-location information to perform complex tasks which interact with their environment. To address this problem, this paper presents a novel method of self-localization based on a combination of extended Kalman filters and Markov methods, obtaining the best features of both methods. This method combines the accuracy and robustness of the Kalman filter and multi-modality of a probabilistic grid. The probabilistic grid generates and validates a dynamic population of extended Kalman filters which compete to represent accurately the robot location. The novelty of this system is to obtain a multimodal estimation of robot location with a simple and effective way for generating, fusing, removing and rating extended Kalman filters. We demonstrate the effectiveness and correctness of this approach in maintaining the position of a humanoid robot in the robot soccer environment, being also suitable for other environments.**

I. INTRODUCTION

We are interested in developing intelligent behaviors for autonomous mobile robots in a variety of scenarios. Many of these behaviors require reliable information on the robot location. The state of the art in self-localization is quite advanced and many of the efforts are refocusing on solving the specific problem of simultaneous localization and mapping (SLAM), in which environment knowledge is built while the robot is positioned in the environment. However, we believe that there is still much work to do in the general case, where the environment is known apriori. Proof that this problem is still active can be seen in soccer humanoid robots [1]. The location information is critical for the robot to make good decisions. Robots fall, are pushed and displaced during operation. Odometry information is not reliable and the main sensor, a camera, faces problems such as occlusions and false positives. Additionally, the game is so dynamic that the response times of each of the modules of the system must be extremely short. It is not easy for a system of self-localization to work properly under these conditions.

We provide a new approach that meets the requirements previously planted as an alternative to the predominant solution in this environment, Monte Carlo Localization (MCL) [2]. The extended Kalman filter, a nonlinear version of the Kalman filter [3], is a good solution for the problem of state estimation, in particular for self-localization. It is robust and fast, but is adequate to maintain the robot location once known, i.e, for a tracking of the robot location. Its weakness is its initialization and face kidnapping situations. Simply it does not work in these cases. This method does not allows to maintain inherently multi-modality in the estimation of the robot location. There are many efforts to achieve a correct and elegant mathematical formulation of the multimodality of these filters, so far unsuccessfully. In contrast, our approach is based on maintaining the current mathematical formulation and generate a population of independent extended Kalman filters. Each of these filters are updated independently with the same sensory information. This solve the problem of multi-modality, but we still have to solve the problem of initialization.

To address this problem, we divided the environment into a probabilistic coarse grain grid, large enough to avoid performance problems and small enough to be useful for our purposes. Each of the cells of the grid has an associated probability which is updated using the available sensory information in a Markovian way [4]. The information grid enables us to define new locations in the environment where it is appropriate to initialize new extended Kalman filter. Finally, based on the grid information and the uncertainty of each of the filters we obtain the robot location. Our approach has many features that make it suitable for highly dynamic environments with real-time requirements. First, extended Kalman filters are suitable for intervals in which the only available information is the displacement of the robot. The uncertainty in the estimate represents the absence of sensory information, but there will still be an estimation about the robot location. Secondly, the initialization is effective at the start of the operation of the robot and after kidnappings, collisions or falls. After incorporating sensory information to the grid, the probability of the cell where the robot actually is rises. If there are no filters previously initialized to represent this new situation, it creates a new one in this cell. The previous existing filters are maintained to deal with situations of false positives. Finally, the performance of the system are suitable for systems with limited computing capacity. The population extended Kalman filter is limited to a small number, and the coarse grained grid also makes the number of cells very low.

An important aspect of this work is that it must operate in real time on the actual humanoid robot. The timing requirements are important when limited computing time has to be shared with other processes such as perception, calculations for the actuators, coordination, behavior generation, and so on. The software architecture used, BICA[5], implements graceful degradation to face heavy load situations. As we have real-time requirements, this mechanism can solve specific problems, but it is important that strict time requirements are met by all processes. Therefore, these requirements have great impact on the design of this system of self-location.

ISBN: 978-84-695-3472-4

The experimental results carried out on the actual robot show that our approach produces reliable self-location information in case of occlusion, collisions and false positives. Although to date this approach has been used successfully in several editions of the RoboCup robot soccer championship, our work opens up new applications in environments where the robot coexists with humans, such as offices and hospitals.

This paper is structured as follows. After discussing related work in the followings section, we will present our extended kalman filter population approach to mobile robot self-localization in section III. In section IV we will present experimental results demonstrating the advantages of our algorithm. Finally, in section V we will discuss the results of the present approach and the possible improvements to be done.

II. Related work

Extended kalman filter is a widely used method of estimating the state of a process. The system state is represented as a Gaussian distribution, with an associated uncertainty. This system has been applied in the location of a mobile robot in several works [6][7]. All of them seeks to address the problem of initialization from the total ignorance, but the solutions are not definitive. We provide a solution to this problem by relying on a probabilistic grid. To determine whether an extended Kalman filter is correct, we use the information in this grid, as well as the uncertainty associated with the filter.

Markov methods have traditionally been successfully used in self-localization of mobile robots. These methods are based on discretizing the environment in a set of states where the robot can find, and keep the probability of being in each of these states. States may have a regular size, building a grid [4], or have topological meaning [8][9]. Typically one location contains several states to represent the orientation of the robot. This method is simple and very flexible to the requirements of many applications, integrating robot self-localization and actuation using partially observable markov decision process models (POMDPS) [10]. However, the major problem of this approach arises when the set of states is high because the environment is too large or because the application requires high precision. some researchers have developed methods to overcome this performance problem trying to reduce the space of states. In the environment of the robot soccer, in [11] is proposed that the orientation is not encoded as a set of states, but as a fuzzy variable, reducing computing needs. Although these requirements are reduced, it still remains a compromise between performance and accuracy. Our approach uses a probability grid is updated using markovian algorithms. The aim of this grid is not to obtain the final robot location, but to point out the locations where starting a new filter extended kalman, or where discard hypothesis. For this reason, we have not performance problems because the grid size is large and thus state space reduced.

More recent work in self-localization of mobile robots are based on particle filters [12], also called Monte Carlo methods [13]. Particle filters metric determine the robot location by sampling the state space where the robot can be. Each particle represents a possible robot location with an associated weight. The weight of each particle varies depending on the robot's sensory information. Those particles with low weights are replaced by others close to those with more weight. The accumulation of particles in a certain location determines where the robot is. This approach is attractive because it does not depends on the the size of the environment and it is able to recover from initial unknowledge or kidnappings, and therefore it is used for many applications [14][15][16]. In particular, some works in soccer robotic bring new features to this method. In [17], it takes advantage of the existing mark locations in the environment to better suit kidnappings and restarts. Lines are also used in [18], in which this algorithm is tuned for an excellent solution, highly adapted to this environment. This method, however, has some problems when there are symmetries in the environment can often decant for the wrong choice prematurely. Furthermore, although able to recover from kidnappings, the dynamics of the particles can make recovery take longer than desired. In contrast, our approach assumptions about the robot location are not dependent on the accumulation of sampling, but complete Gaussian estimations. This allows previously discarded hypothesis can be re-considered. In addition, different symmetrical estimations can be maintained without discarding any hypothesis until to end up with symmetry. Recoveries from kidnapping are faster and do not depend on creating a new cluster of samples, but starting a new Kalman filter extended to the new location.

III. Self-localization method

The approach proposed in this work tries to self-localize the robot in its environment. The main information source is the one extracted by the images taken from the robot's camera. It also uses the information that provides the robot framework related to the movement and the motor odometry.

The robot is in a real 3D world, with three values to represent the location, and another three to represent the orientation. We assume that the robot is (or should be) always in vertical position, and with any foot on the floor. So, the robot location can be simplified to three values (x, y, θ), taking the robot location as a point on the floor, between both legs. Then, It represents the robot location, in 2D ($z = 0$), and a orientation with respect the Z axis.

To illustrate the concepts more clearly, we will focus on the environment of soccer robots. This environment has a size of 6×4 meters, and has a set of visual characteristics (color goals and field lines) whose location is known a priori. Still, the method proposed in this paper is applicable to any environment where there is a set of elements whose location is known a priori, and a set of lines that can be visually detectable (junction between walls and floor, for example).

III-A. Individual extended Kalman filter

Extended kalman filter is an effective method to estimate the state of a process. This method has been used successfu-

ISBN: 978-84-695-3472-4

lly to self-localize a mobile robots, but mainly for tracking, ie this is a local method. The state we want to estimate the robot location defined as the state vector $s = \begin{pmatrix} x & y & \theta \end{pmatrix}^T$ and its uncertainty P. 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 σ^u_{t-1}, to the present state s_t, according to Figure III-A.1.

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

Each component of s_t is calculated independently using the following decomposition of f:

$$x_t = x_{t-1} + (u^x_{t-1} + \sigma^x_{t-1})cos\theta_{t-1} - (u^y_{t-1} + \sigma^y_{t-1})sen\theta_{t-1} \tag{2}$$

$$y_t = y_{t-1} + (u^x_{t-1} + \sigma^x_{t-1})sen\theta_{t-1} + (u^y_{t-1} + \sigma^y_{t-1})cos\theta_{t-1} \tag{3}$$

$$\theta_t = \theta_{t-1} + u^\theta_{t-1} + \sigma^\theta_{t-1} \tag{4}$$

III-A.1. Prediction step: In this step we calculate s_t^- and P_t^-. s_t^- is the location in which the robot will be according to its previous location and the information of the odometry using the movement model, which is based on the displacement of the robot calculated by the motion module. Left side in Figure III-A.1 shows the information related to a displacement. This information is the displacement vector $u = (u_x, u_y, u_\theta)$ and its associated uncertainty $\sigma_u = (\sigma^x_u, \sigma^y_u, \sigma^\theta_u)$.

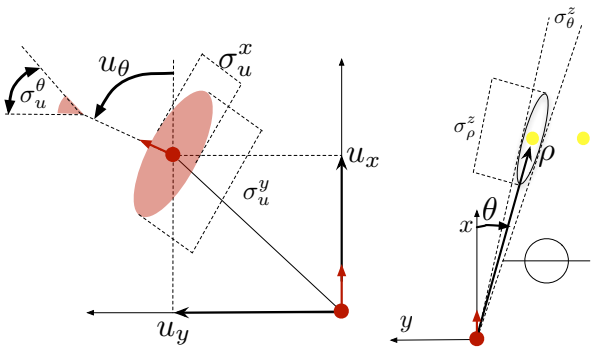

Fig. 1. Odometry based movement model and observation model

$$s_t^- = \begin{pmatrix} x_t^- & y_t^- & \theta_t^- \end{pmatrix}^T = f(s_{t-1}, u_{t-1}, 0) \tag{5}$$

$$P_t^- = A_t P_{t-1} A_t^T + W_t Q_{t-1} W_t^T \tag{6}$$

Where P_{t-1} updates to $A_t P_{t-1} A_t^T$ for representing the previous noise after control input u (Figure 2) and Q_t is the noise in the prediction process. $W_t Q_{t-1} W_t^T$ represents the noise Q added to P_t^- from P_{t-1}. Applying transformation matrix W_t(equation 9), we add this noise to the uncertainty.

$$A_t = \frac{\partial f}{\partial s} = \begin{pmatrix} 1 & 0 & -u^y_{t-1}cos\theta_{t-1} - u^x_{t-1}sen\theta_{t-1} \\ 0 & 1 & u^x_{t-1}cos\theta_{t-1} - u^y_{t-1}sen\theta_{t-1} \\ 0 & 0 & 1 \end{pmatrix} \tag{7}$$

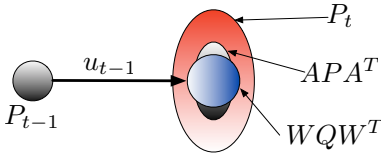

Fig. 2. Combination of the new uncertainty P_t^- from the prior P_{t-1}

$$Q_t = E[\sigma_t \sigma_t^T] = \begin{pmatrix} (\sigma^x_u)^2 & 0 & 0 \\ 0 & (\sigma^y_u)^2 & 0 \\ 0 & 0 & (\sigma^\theta_u)^2 \end{pmatrix} \tag{8}$$

$$W_t = \frac{\partial f}{\partial \sigma_u} = \begin{pmatrix} cos\theta_{t-1} & -sen\theta_{t-1} & 0 \\ sen\theta_{t-1} & cos\theta_{t-1} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{9}$$

III-A.2. Update step: The robot detects the visual landmarks of the environment using the images from its camera. In this work, the relevant landmarks are the goal posts and the center of the field, as shown at right side in Figure III-A.1.Each image has an associated information of the angles of each of the actuators of the robot. This information is used to perform 2D to 3D transformations of image points to real-world coordinates. We do not want to go into detail about the image processing because we do not want to distract from the central part of this work, so we briefly describe this process. First, we filter and segment the image depending on the color of each pixel. From groups with similar color, we detect the field and the candidates for visual landmarks: goalposts and field lines. Finally, we convert the image space to real-world space to check the dimensions and locations of each of these candidates. Those with valid properties are definitely used. We do not use instant perceptions, but an estimate of each using Joint Probabilistic Data Association Filter (JPDAF) [19]. This filter updates a estimation each landmarks, which are represented as $z = (\rho, \theta)$ in polar coordinates with respect the robot, with an associated uncertainty $\sigma_z = (\sigma^z_\rho, \sigma^z_\theta)$, as shown at right side in Figure III-A.1.

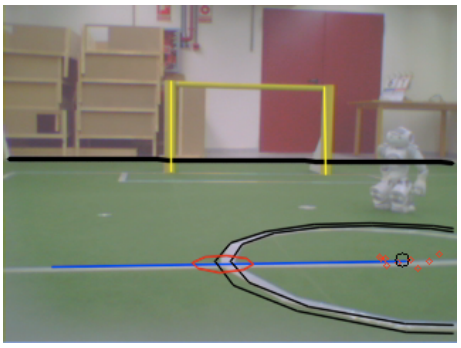

Fig. 3. Visual landmarks for the self-localization method.

The uncertainty associated with both the movement and the perceptions is empirically calculated. After intensive testing, we can measure the accuracy of the information from the camera and the module that calculates the movement of the robot from the parameters of the walk. The uncertainty

ISBN: 978-84-695-3472-4

is high compared to a wheeled robot because our robot is a biped and accuracy in the calculation of the position of the camera depends on all the joint sensors from the floor to the robot head. We update the robot state using the observations z_i related to the landmark m_i.

$$s_t = s_{t-1} + K_t^i(z_t^i - \hat{z}_t^i) = s_{t-1} + K_t^i(z_t^i - h^i(s_{t-1})) \quad (10)$$

$$P_t = (I - K_t^i H_t^i)P_{t-1}. \quad (11)$$

where $h(s_{t-1})$ is a nonlinear function that give us the theoretical observation of z in the state s_{t-1}.

$$\hat{z}_t^i = h^i(s_{t-1}) = \begin{pmatrix} \sqrt{(m_{t,x}^i - s_{t-1,x})^2 + (m_{t,y}^i - s_{t-1,y})} \\ atan2(m_{t,x}^i - s_{t-1,x}, m_{t,y}^i - s_{t-1,y}) - s_{t-1,\theta} \end{pmatrix} \quad (12)$$

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

$$K_t^i = P_{t-1}(H_t^i)^T(S_t^i)^{-1} \quad (13)$$

$$S_t^i = H_t^i P_{t-1}(H_t^i)^T + R_t^i \quad (14)$$

$$H_t^i = \frac{\partial h^i(s_{t-1})}{\partial s_t} = \begin{pmatrix} -\frac{m_{t,x}^i - s_{t-1,x}}{\sqrt{q}} & -\frac{m_{t,y}^i - s_{t-1,y}}{\sqrt{q}} & 0 \\ \frac{m_{t,y}^i - s_{t-1,y}}{q} & -\frac{m_{t,x}^i - s_{t-1,x}}{q} & -1 \\ 0 & 0 & 0 \end{pmatrix} \quad (15)$$

$$q = (m_{t,x}^i - s_{t-1,x})^2 + (m_{t,y}^i - s_{t-1,y})^2 \quad (16)$$

$$R_t^i = E[\sigma^z(\sigma^z)^T] = \begin{pmatrix} (\sigma_\rho^z)^2 & 0 \\ 0 & (\sigma_\theta^z)^2 \end{pmatrix} \quad (17)$$

III-B. Markovian grid

We represent the robot's environment as a regular grid. The cell size is configurable, setting it to 50 cm wide. A environment which size is 6×4 meter is represented with a grid composed by 96 cells. Since it is a regular grid, is easy to calculate from metric coordinates (x, y) to the coordinates (i, j) in the grid. To set some terminology, the grid is a set of states S. Each state $s_{i,j} \in S$ corresponds to a specific cell. $Bel(S = s_{i,j})$ is the probability of being at state $s_{i,j}$, which can be also represented as $p(s_{i,j})$. At each state $s_{i,j}$, the movement model defines a $p(s_{i,j}|s'_{m,n}, u)$ for all $s, s' \in S$ using the robot displacement u. The observation model defines $p(z|s_{i,j})$ as the probability of perceiving the visual feature z at $s_{i,j}$. These models are calculated from the models presented in the previous section, which updates the probability distribution over all the states using equation 18 and 19.

- When the robot moves, the movement u updates each cell probability using the movement model $p(s_{i,j}|s'_{m,n}, u)$.

$$p_{post}(s_{i,j}) = \alpha \times \sum_{s'_{m,n} \in S} p(s_{i,j}|s'_{m,n}, u) \times p_{prior}(s_{i,j}) \quad (18)$$

- When the robot perceives a visual feature z, each cell is updated using the observation model $p(z|s_{i,j})$.

$$p_{post}(s_{i,j}) = \alpha \times p(z|s_{i,j}) \times p_{prior}(s_{i,j}) \quad (19)$$

where α is a normalization factor to ensure that the probabilities all sum to one.

Most of markov models define several states at each location to represent the orientation of the robot. This is quite expensive, as it can multiply the computation time required depending on the desired resolution. Instead of using several states in each cell, our solution consists in representing the orientation as a Gaussian distribution in the range $[-\pi, \pi]$. This distribution is updated with the sensory information z and the robot rotation u_θ. We name $\Theta(s_{i,j})$ as the mean of this distribution, and $\sigma_\Theta(s_{i,j})$ the standard deviation.

- When the robot moves, the component of rotation of the movement u_θ updates the distribution $N(\Theta(s_{i,j}), \sigma_\Theta(s_{i,j}))$.

$$\Theta^{post}(s_{i,j}) = \Theta^{prior}(s_{i,j}) + u_\theta \quad (20)$$

$$\sigma_\Theta^{post}(s_{i,j}) = \sigma_\Theta^{prior}(s_{i,j}) + \sigma_{u_\theta} \quad (21)$$

- When the robot perceives a visual feature z, it updates the distribution. In this equations, $h(s_{i,j})$ is a function that give us the theoretical observation of z in the state $s_{i,j}$.

$$\Theta^{post}(s_{i,j}) = \Theta^{prior}(s_{i,j}) + \frac{\sigma_\Theta}{\sigma_\Theta + \sigma_z}(z - h(s_{i,j})) \quad (22)$$

$$\sigma_\Theta^{post}(s_{i,j}) = \frac{\sigma_\Theta}{\sigma_\Theta + \sigma_z}\sigma_\Theta^{prior}(s_{i,j}) \quad (23)$$

III-C. Population dynamics

The real value of this work is how to design a method of self-localization effective from the best characteristics of the elements previously described. The grid of probability is an global self-localization method that allows multimodality and error recovery. As a weaknesses, it lacks of enough robustness, and it is necessary to choose between accuracy and performance. Extended kalman filter is a lightweight and robust method, but it lacks of multimodality and recovery from errors.

The central idea is to control the population dynamics of extended kalman filter. This population grows and reduces using information Markovian grid. Once an extended Kalman filter is created, it runs in parallel with the rest of the filters in the population. Each filter independently, incorporates information from sensory perception and movement. Figure 4 shows the overall system. The robot position is selected among the filters taking into account the markovian grid information, which is also used for creating new filters. The population is updated using three rules: creation, destruction and combination:

- **Creation**. This rule is executed when the Markovian grid indicates that the robot is likely to be in any

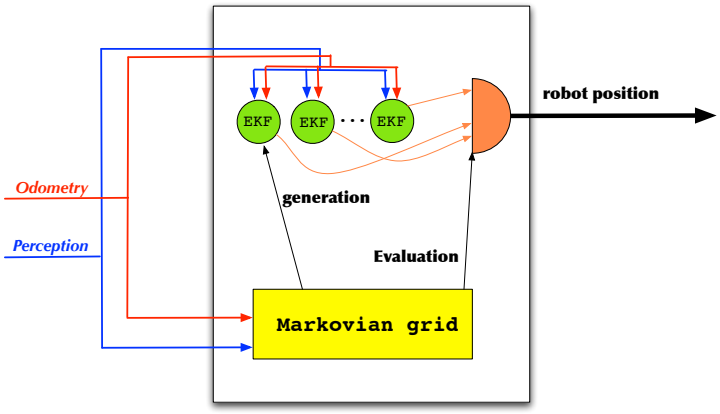

Fig. 4. Diagram of the overall system.

location that is not covered by any filter forming part of the population. This happens at the start of robot operation or after a kidnapping.

- **Destruction**. There are two factors affecting the destruction of an extended Kalman filter. First, the uncertainty of the filter has overgrown because it does not incorporate perceptions consistent with its status. Second, the probability of the cell corresponding to the location of the filter is very low.
- **Combination**. If two filters of the population converge to the same location and orientation, we remove the filter with higher uncertainty. This rule makes lower the workload and eliminates expendable filters.

The algorithm which describes how to apply these operations is shown in the Algorithm 1. Lines 11-17 define the **combination** rule, lines 18-22 define the **destruction** rule and lines 23-29 define the **creation** rule. We choose the best estimate of the extended Kalman filter of the population as the robot location. This choice is made based on the probability of the grid cell in which it is, and its similarity to the orientation that is in the cell, which also defines the reliability of this information.

Algorithm 1: Combination of a Markovian grid and a population of extended Kalman filters.

```
1  Initialize grid using full uncertainly;
2  while true do
3      Predict grid using using motion;
4      foreach ekf_i do
5          Predict ekf_i using motion;
6      end
7      Correct grid using landmark information;
8      foreach ekf_i do
9          Correct ekf_i using landmark information;
10     end
11     foreach ekf_i do
12         foreach active ekf_j do
13             if distance(ekf_i, ekf_j) is near) and angle(ekf_i, ekf_j)
               is similar then
14                 ekf_m = new ekf from ekf_i and ekf_j combination
15             end
16         end
17     end
18     foreach ekf_i do
19         if grid(ekf_i.x, ekf_i.y).prob() is low and ekf_i.uncertainty
           is high then
20             remove ekf_i;
21         end
22     end
23     if (grid(x, y).prob() is high) then
24         foreach ekf_i do
25             if (distance(ekf_i, grid(x, y)) is far) then
26                 ekf_m = new ekf from grid(x, y);
27             end
28         end
29     end
30 end
```

IV. Experimentation

The approach presented in this paper is intended to be robust, efficient and reliable. It is necessary to provide experimental evidence that this method meets these expectations. Although this method is intended to be general and not focused on a robot and a particular environment, we must set the environment and platform that will be used in the experimental stage. The experimental platform is the Nao humanoid robot. This robot has become very popular in recent years due to its low cost and its advanced features. It is 55 cm tall, has 25 dof and the main sensor is a camera on his head. The test environment is the real field of robot soccer shown in Figure III-A.1. We are working in ground truth system for an accurate test and comparison with other self-localization methods. Unfortunately, it was not ready for using it in these tests. We hope that the experimental results are good enough to provide evidence of the correct operation of this system.

The first experimental result concerns to the robustness and accuracy of this method. This experiment has carried out by making the robot walk on the field for several seconds. After this time, we stop the robot and compare the estimation method of self-localization with the actual robot location, measured manually. Each iteration of this test takes about 30 seconds in which the robot moves forward, sideways and rotates. We have performed 15 trials of this test. After this experiment, the average error in estimating the location is 23 cm, with a standard deviation of 10 cm.

The second experiment measures the time it takes the robot to recover after a kidnapping. During this experiment, the robot is stopped, but his head moves in search of visual stimuli. Once self-localized, the robot is moved manually to another location in the field. We repeated this operation 15 times and we measured the time the robot takes to recover (when the location error is less than 20 cm and the orientation is less than 0.2 radians). The results show that the robot spends on recovery 8.3 seconds, on average, with a standard deviation of 7 seconds. In all cases, the previous estimate continues to exist, but their uncertainty increases and the probability of the lower cell at that location. If there had been no kidnapping and perception was due to a false positive, the recovery would be very efficient when the false positive disappeared.

Finally, we have collected information from the computation time of each module involved in a real game situation. During this situation the robot must perceive the ball and the goal, be self-localized, generate behaviors, use visual attention for controlling the camera position and generate

ISBN: 978-84-695-3472-4

Module	Median	Average	Stdev
Actuation	0.120 ms	0.458 ms	0.017ms
Perception	9.34 ms	7.3087 ms	11.995 ms
Behaviors	0.031 ms	0.173 ms	1.030 ms
Visual Attention	0.111 ms	0.541 ms	1.7304 ms
Self-localization	0.364 ms	1.736 ms	3.0762 ms

TABLE I

COMPUTATION TIMES FOR THE MODULES

the commands for robot's actuators. The table I contains the results. In a system that obtains images at 30Hz, 1.736 ms on average in self-localization is very low computation consumption. Compared with the time spent on perception, 7.3 ms on average, is a quite acceptable time.

V. FUTURE WORK AND CONCLUSIONS

This paper has presented a method of visual self-localization of autonomous mobile humanoid robots. This method allows to maintain a dynamic population of extended Kalman filters that grows or decreases using a grid Markov info. The benefits of this method is to take advantage of the best characteristics of Gaussian and Markovian methods. Extended Kalman filters are robust and lightweight. A population of Kalman filters provides multimodality, extending this type of algorithms. The markovian grid provides hypotheses which create or discard filters in this population. Under these principles, each possible robot location is covered with a different extended Kalman filter. The system recovers from kidnapping situations because the markovian grid generate new hypothesis as soon as the robot perceives new features. Finally, it is robust to false positives because it does not immediately discard the correct locations which are temporary not corroborated perceptively.

This approach has been tested in the real robot in the robot soccer environment, but is likely to be used in a variety of environments. Preliminary experiments are promising, but it must be tested more extensively with a ground truth system and compared with relevant algorithms (Monte Carlo and Markov, for example). Still, it has shown its recover ability, its accuracy and low computational cost.

We are extending this work in several directions. First, we are evaluating using field lines as a new perception. These lines are natural to other environments such as offices or hospitals. This is related to another possible direction for this work: leave the field of robotic soccer and get into the real world, where the capacity for self-localization is important in a wide variety of applications. Finally, we are working on the detection of other robots to collaborate in self-location. If the robot A detects the robot B, we can create new hypotheses on the robot B to improve its self-localization.

VI. ACKNOWLEDGEMENTS

This work has been supported by the project S2009/DPI-1559, RoboCity2030-II, from the Comunidad de Madrid and by the project 10/02567 from the Spanish Ministry of Science and Innovation.

REFERENCES

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

[2] Frank Dellaert, Dieter Fox, Wolfram Burgard and Sebastian Thrun, *Monte carlo localization for mobile robots*. In IEEE International Conference on Robotics and Automation (ICRA99), pp. 1322–1328, (1999).

[3] Rudolph E. Kalman, *A New Approach to Linear Filtering and Prediction Problems*. Transactions of the ASME, Journal of Basic Engineering, Vol. 82, No. Series D, pp. 34?45, (1960).

[4] Dieter Fox, Wolfram Burgard and Sebastian Thrun, *Markov Localization for Mobile Robots in Dynamic Environments*, Journal of Artificial Intelligence Research, Vol. 11, pp. 391–427, 1999.

[5] Carlos Agüero, José M. Cañas, Francisco Martín and Eduardo Perdices, *Behavior-based Iterative Component Architecture for soccer applications with the Nao humanoid*, Proceedings of the 5th Workshop on Humanoid Soccer Robots, inside 10th IEEE-RAS Int. Conf. Humanoid Robots. Nashville, TN, (2010)

[6] Evgeni Kyriy and Martin Buehler, *Three-state Extended Kalman Filter for Mobile Robot Localization*, Carnegie Mellon Technical Report, Pittsburgh, PA(2002)

[7] Lastra, R., Vallejos, P. and Ruiz-del-Solar, J, *Self-Localization and Ball Tracking for the RoboCup 4-Legged League*, Proceeding of the 2nd IEEE Latin American Robotics Symposium LARS 2005, Santiago de Chile, Chile (2005)

[8] Sebastian Thrun and Arno Bücken, *Integrating Grid-Based and Topological Maps for Mobile Robot Navigation*, Proceedings of the AAAI Thirteenth National Conference on Artificial Intelligence, Vol. 2, pp. 944–950. Portland, OG(1996)

[9] Francisco Martín, Vicente Matellán, José María Cañas and Pablo Barrera, *Visual Based Localization for a Legged Robot*, Lecture Notes on Computer Science. Vol. LNAI-4020. pp 708–715. (2006).

[10] Sven Koenig and Reid Simmons, *Xavier: A Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models*, Artificial Intelligence Based Mobile Robotics: Case Studies of Successful Robot Systems, MIT Press, pp. 91–122 (1998).

[11] P. Buschka, A. Saffiotti and Z.Wasik, *Fuzzy Landmark-Based Localization for a Legged Robot*, Proceedings of the International Conference on Intelligent Robots and Systems 2000, pp 1205–1210, Takamatsu, Japan (2000)

[12] Sebastian Thrun, *Particle Filters in Robotics*, Proceedings of the 18th Annual Conference on Uncertainty in Artificial Intelligence (UAI-02), pp. 511-518. San Francisco, CA (2002).

[13] Dieter Fox, Wolfram Burgard, Frank Dellaert, Sebastian Thrun, *Monte Carlo Localization: Efficient Position Estimation for Mobile Robots*, Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99), pp. 343-349. Orlando, FL. (1999).

[14] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schultea and D. Schulz, *Probabilistic Algorithms and the Interactive Museum Tour-Guide Robot Minerva*, International Journal of Robotics Research, Vol. 19, No. 11. (2000), pp. 972-999.

[15] R. Conde, A. Ollero and J.A. Cobano, *Method based on a particle filter for UAV trajectory prediction under uncertainties*. 40th International Symposium of Robotics. Barcelona, Spain (2009).

[16] Nak Yong Ko, Tae Gyun Kim and Sung Woo Noh, *Monte Carlo Localization of Underwater Robot Using Internal and External Information*. 2011 IEEE Asia-Pacific Services Computing Conference, APSCC. 2011, pp. 410–415. Jeju, South Korea (2011).

[17] Scott Lenser and Manuela Veloso, *Sensor Resetting Localization for Poorly Modelled Mobile Robots*, Proceedings of ICRA-2000, the International Conference on Robotics and Automation, pp. 1225-1232. San Francisco, CA (2000).

[18] T. Röfer, T. Laue and D. Thomas, *Particle-filter-based self-localization using landmarks and directed lines*, RoboCup 2005: Robot Soccer World Cup IX, Vol. 4020, pp. 608?615, Lecture Notes in Artificial Intelligence. Springer (2006).

[19] Daronkolaei, Aliakbar Gorji, Shiry, Saeed, Menhaja and Mohammad Bagher, *Multiple Target Tracking for Mobile Robots Using the JPDAF Algorithm*, Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence, Vol 01, pp. 137–145. Washington, DC (2007)