Vision-based Monte Carlo - Kalman Localization in a Known Dynamic Environment

Xiaohan Zhang and Xiaoping Chen
Department of Computer Science
Univ. of Sci. & Tech. of China
Hefei, Anhui 230027, China
Email: zxhans, xpchen@mail.ustc.edu.cn

Jialing Li
Department of ECE
Polytechnic University
Brooklyn, NY, 11201, USA
Email: jli16@utopia.poly.edu

Xiang Li
ZTE Research Center, ZTE Corporation
Nanjing, Jiangsu,
210012 China
Email: li.xiang2@zte.com.cn

***Abstract*— Localization is one of the fundamental problems in mobile robot navigation. In this paper, we present a vision-based localization method called Monte Carlo - Kalman localization (MCL-EKF). This method is a combination of Monte Carlo localization (MCL) and Extended Kalman Filter (EKF) enhancement. We firstly give a detailed implementation of MCL with the emphasis on dealing with multiple types of perceptual information and solving the problem of robot kidnapping. Next, we establish EKFs on landmarks to build a real-time environment around the robot. Information from this real-time environment will be utilized by the perception model of MCL. We also elaborate on our methods of dealing with a single or two landmarks in the perception model. We carry out all experiments on Sony AIBO ERS-7 robots. Results show that the MCL-EKF reduces perceptual errors, increases precision and stability and still keeps a good ability of recovery.**

Keywords: Localization, Monte Carlo Localization, Extended Kalman Filter, Mobile Robots.

I. Introduction

Leonard et al. [1] summarized the general problem of mobile robot navigation by three questions: "Where am I?", "Where am I going?", and "How should I get there?". The problem of robot *localization* consists of answering the first question from a robot's point of view, that is, to find out its position relative to the environment. Monte Carlo Localization (MCL) [3] and Kalman Filters [4] are two popular kinds of localization methods.

Monte Carlo Localization is a probabilistic approach that has been shown to be a robust robot localization method, especially in dynamic environments [5], to solve the "kidnapped robot" problem. It is reported in [6] that MCL provides less accurate results than some other localization methods (e.g. the Extended Kalman Filters) under certain conditions. In our implementation environment, where, under large perceptual uncertainties, high accuracy and stability[1] of localization results are needed, we find it difficult to achieve the goal using MCL only.

Many previous researches on MCL (e.g. [3, 7]) have used distance sensors such as laser scanners or sonar sensors. Only in a few recent approaches [8, 9], is directed robot vision used in MCL. These vision-based approaches are commonly based on landmark detection. Landmarks are distinct features that a robot can recognize in a known environment from its sensory input. As described in [2], landmark navigation is rather inaccurate when the robot is further away from the landmark. Sridharan et al. [10] presents a "distance-based updates" enhancement, collecting data in the real environment and making corrections through offline training. Although practically possible, it relies on an environment with fixed property, cannot adjust itself in real-time and can only change landmark's distance.

On the other hand, the Extended Kalman Filter (EKF) [13] is a recursive data processing algorithm that estimates the state of a noisy dynamic system. It has been used in a wide range of applications including mobile robot localization, in which it encountered some difficulties in solving global localization and robot kidnapping problems [4]. Furthermore. most of the previous researches (e.g. [4, 6, 12]) focus on tracking and estimating the position of the robot, few is paying attention to build the environment and to enhance and combine with other localization algorithms like MCL.

The basic idea of this paper is to establish Extended Kalman Filters to model the real-time environment and diminish perceptual errors, then make use of the filtered perceptual information for Monte Carlo Localization. In this way, we inherit the property of MCL on robustness and recovery from manual robot displacement, and the property of EKF on precision and stability. In the next section, we describe a detailed implementation of MCL. In Section III, we demonstrate how EKFs work out for Monte Carlo - Kalman Localization. We analyze a detailed perception model on a single or two landmarks in Section IV. In Section V, we investigate the uncertainties in the environment and conduct a set of experiments on AIBO [14] robots to testify and compare the performance of localization systems. Finally, we conclude with Section VI.

II. A Detailed Implementation of Monte Carlo Localization

A. Basic Monte Carlo Localization

Monte Carlo Localization is a discrete form of Markov localization [5]. In MCL, the robot has a belief about where it is. The belief is represented by a set of weighted, random particles or samples. A sample set constitutes a discrete

[1]The ability of a robot to stay localized at the desired position once it has arrived. See Section V-C for details.

approximation of a probability distribution. Samples in MCL are of the type $((x, y, \theta)^T, p)$, where x, y, θ denote a robot's position and orientation, and p is a numerical weighting factor, analogous to a discrete probability.

In analogy with the general Markov localization approach, MCL firstly proceeds in two phases, namely the action model and the perception model. After that, a so-called resampling process is introduced to generate samples from the sample set according to their *p*-values. Finally, the average probability distribution of current robot pose is determined, and the process repeats recursively from the beginning.

We present most of the steps in this section. We will discuss perception model in details in section IV.

B. Action Model

When the robot performs an action, assume the odometry data (in the global reference frame) is $(dx, dy, d\theta)^T$. For each sample, the new position and orientation is calculated as follows:

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} dx \\ dy \\ d\theta \end{pmatrix} + \begin{pmatrix} gx \\ gy \\ g\theta \end{pmatrix} \tag{1}$$

where $(gx, gy, g\theta)^T$ represents three dimensions' zero-mean Gaussian noise. It simulates the errors of action model. *P*-value of each sample remains the same without the resampling process, because odometry data are not sufficient to form a filtering rule for selecting samples and this simplification will not influence the final localization result.

C. General Perception Model and Calculating the P-value

We can obtain many types of perceptual information, such as a single landmark's distance or angle observed relative to the robot, and a pair of landmarks' distance or angle. Usually, we can derive single or multiple types of information from each image frame. We will describe methods to calculate *p*-value for each type of perception update in section IV. Assume that there are M types and p_i denotes the *p*-value of the *i*th type. To make the most of perceptual information, for each sample $((x, y, \theta)^T, p)$, we have

$$p = min(p_i), i = 1, 2, \ldots, M. \tag{2}$$

i.e., the final *p*-value is the minimum. Through controlling the calculation of each p_i, we can control the effect that each type of perceptual information brings. For example, distance information of a distant landmark should have less influence on the sample set than that of a close landmark. Less reliable information, such as distance information of landmark, should place less effect than more reliable information, such as angle information of landmarks. With such strict constraints, we can choose the most influential type of information to calculate the *p*-value and lay the foundation of resampling process.

D. The Process of Resampling

First, samples in the sample set are copied into an assistant sample set according to their *p*-values. The higher *p*-value a sample has, the higher the probability that it would be copied. In contrast, the lower *p*-value it has, the lower the probability that it would be copied; and if the *p*-value is too small, the sample might not be copied. Consequently, the assistant sample set agrees with the probability distribution that perceptual information suggests.

Next, *p*-values of samples in the assistant sample set will be normalized. The three dimensions of a sample will add up small offsets to diffuse their values around their original values. After that, the assistant sample set replaces the original one to be a new sample set. The whole formula is as follows:

$$\begin{aligned} \begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} &= \begin{pmatrix} \sigma_{xy} \times (1-p) \times rndGauss \times LCount \\ \sigma_{xy} \times (1-p) \times rndGauss \times LCount \\ \sigma_{angle} \times (1-p) \times rndGauss \times LCount \end{pmatrix} \\ &\quad + \frac{1}{3} \begin{pmatrix} \sigma_{xy} \times rndGauss \\ \sigma_{xy} \times rndGauss \\ \sigma_{angle} \times rndGauss \end{pmatrix} + \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} \end{aligned} \tag{3}$$

where $rndGauss$ is a function that generates a normal random variable. Constants σ_{xy} and σ_{angle} are diffusion ranges of corresponding dimensions. $LCount$ is a variable counting the number of times that consecutive large errors of perceptual information have occurred, and accelerates the diffusion process in that case. The diffusion process can be deemed as a proper way to obtain estimates of the distribution density of robot's actual location. In the next resampling period, certain diffused samples, which are closer to the robot's actual position, would be assigned higher *p*-value, hence, would be more probable to be copied into the assistant sample set in the first step of that period.

In this manner, when temporary errors in perceptual information occur, the sample set diffuses temporarily and thus bad influence on the final localization result can be reduced. On the other hand, when the final localization result comes across big errors due to robots' congestion or unknown movements (robot kidnapping), the sample set diffuses promptly, and even covers most regions of the environment. As a result, the samples congregates quickly into a more reasonable range in the next resampling period. Given the confidence of the final localization result (see Sec. E), the declining confidence resulted from the diffusion process, will indicate the unreliability of the final result.

E. Estimating the Final Localization Result

Assume that there are *N* samples. After the process of resampling, the final localization result can be given by:

$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix}_{pos} = \begin{pmatrix} \sum_{i=1}^{N} x_i \times p_i \\ \sum_{i=1}^{N} y_i \times p_i \\ \arctan(\sum_{i=1}^{N} \sin\theta_i, \sum_{i=1}^{N} \cos\theta_i) \end{pmatrix}. \tag{4}$$

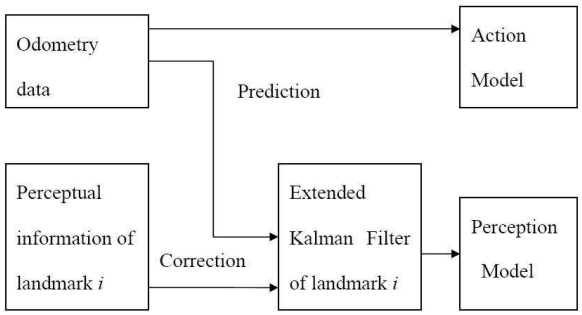

Fig. 1. The core process of MCL-EKF localization system, using EKF to build the environment and to generate information for models of MCL.

Note that averaging the angles is not straightforward, because of their circularity. c_{xy} and c_θ, the confidence of the final position $(x, y)^T$ and orientation θ, respectively, indicating to what extent the samples diffuse, are given by second central moment as follows.

$$c_{xy} = 1 - a_{xy} \sum_{i=1}^{N} \left[(x_i - x_{pos})^2 + (y_i - y_{pos})^2 \right] \tag{5}$$

$$c_\theta = 1 - a_\theta \sum_{i=1}^{N} (\theta_i - \theta_{pos})^2 \tag{6}$$

where $\alpha_{x,y}$ and α_θ are experiential values by which we can restrict $c_{x,y}$ and c_θ to $[0, 1]$.

III. Establishing Extended Kalman Filters for MCL-EKF

On the other hand, the Kalman Filter [13] is a well-known technique for state and parameter estimation. It is a recursive estimation procedure that uses sequential sets of measurements. Prior knowledge of the state (expressed by the covariance matrix) is improved at each step by taking the prior state estimates and new data for the subsequent state estimation. The Kalman Filter requires a linear system model and a linear measurement model, but many dynamic system and sensor models are not completely linear, so the Extended Kalman Filter (EKF) [4, 6] is introduced to deal with nonlinear models. EKF linearizes the nonlinear models by dropping the higher order terms of a Taylor series expansion.

We establish Extended Kalman Filters on each individual landmark and altogether we can build a real-time environment around the robot. The perception model of MCL utilizes information from this real-time environment. See Fig. 1 for details.

A. System Model

In EKF, system model is defined as follows:

$$x'_k = f(x'_{k-1}) + \omega_{k-1} \tag{7}$$

where x'_k describes the filter state at time step *k*, i.e., the landmark position in the robot's reference frame at time step *k*, and $f(\cdot)$ is a nonlinear system function relating the state of the previous time step to the current state. ω_{k-1} represents the noise corrupting the system, assuming to be independent, white, zero-mean, and Gaussian distributed, i.e. $\omega_k \sim N(0, Q_k)$.

Assume the odometry data (in the robot's reference frame) is denoted by $(dx, dy, d\theta)^T$, we can rewrite (7) in details by:

$$\begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} \cos(d\theta) & \sin(d\theta) \\ -\sin(d\theta) & \cos(d\theta) \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} + \begin{pmatrix} dx \\ dy \end{pmatrix} + \omega_{k-1} \tag{8}$$

where $(x_k, y_k)^T$ represents the landmark's coordinate in the relative reference frame at time step *k*. Equation (8) is translating between relative and global reference frames.

B. Measurement Model

In EKF, measurement model is defined as follows:

$$z'_k = h(x'_k) + \upsilon_k \tag{9}$$

where z'_k is the true measurement, x'_k describes the state of landmark, and $h(\cdot)$ is a nonlinear measurement function relating the state of the system to a measurement. is the noise corrupting the measurement υ_k, assuming to be independent, white, zero-mean, and Gaussian distributed, i.e., $\upsilon_k \sim N(0, R_k)$. To maintain the landmarks' state in a robot's reference frame, we should measure the perceptual information of a known landmark including angle and distance to the robot, denoted by variables *angle* and *dist*. Then we can rewrite (9) in details by:

$$\begin{pmatrix} dist \\ angle \end{pmatrix} = \begin{pmatrix} \sqrt{x_k^2 + y_k^2} \\ \arctan(x_k, y_k) \end{pmatrix} + \upsilon_k. \tag{10}$$

C. Initialization

EKF is initialized with the posterior state estimate $(x_0, y_0)^T$ and uncertainty P_0 at time step 0. Since it is difficult for EKF to determine initial states by itself, the initial state of a certain landmark will be determined by the first detection of that landmark, i.e. the perceptual inputs for the first time.

D. Prediction

At every time step, EKF propagates the state and uncertainty of the system at the previous time step to the current time step using the prediction equations:

$$x'_k = f(x'_{k-1}) \tag{11}$$

$$P_k^- = A_k P_{k-1} A_k^T + Q_{k-1} \tag{12}$$

where the Jacobian matrix A_k contains the partial derivatives of system function $f(\cdot)$ with respect to state x' , evaluating at x'_{k-1} of the last time step,

$$A_k = \frac{\partial f(x')}{\partial x'}\Big|_{x'=x'_{k-1}} = \frac{\partial f(x,y)}{\partial (x,y)}\Big|_{x=x_{k-1}, y=y_{k-1}}. \tag{13}$$

In addition, by incorporating (8) into(13), we can obtain:

$$A_k = \begin{pmatrix} \cos d\theta & \sin d\theta \\ -\sin d\theta & \cos d\theta \end{pmatrix}. \tag{14}$$

E. Correction

Here are the correction equations EKF uses:

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + R_k)^{-1} \tag{15}$$

$$x'_k = x'_k + K_k(z_k - h(x'_k)) \tag{16}$$

$$P_k = (1 - K_k H_k) P_k^-. \tag{17}$$

The Jacobian matrix H_k can be obtained from the following equation:

$$\begin{aligned} H_k &= \frac{\partial h(x')}{\partial x'}\Big|_{x'=x'_k} = \frac{\partial f(x,y)}{\partial (x,y)}\Big|_{x=x_k, y=y_k} \\ &= \begin{pmatrix} \frac{\partial dist}{\partial x} & \frac{\partial dist}{\partial y} \\ \frac{\partial angle}{\partial x} & \frac{\partial angle}{\partial y} \end{pmatrix}. \end{aligned} \tag{18}$$

After incorporating (10) into (18), we can finally obtain:

$$H_k = \begin{pmatrix} \frac{x_k\sqrt{x_k^2+y_k^2}}{x_k^2+y_k^2} & \frac{y_k\sqrt{x_k^2+y_k^2}}{x_k^2+y_k^2} \\ -\frac{y_k}{x_k^2+y_k^2} & \frac{x_k}{x_k^2+y_k^2} \end{pmatrix}. \tag{19}$$

The new posterior belief from (16) and (17) is used in the next time step's prediction and the whole process is recursively implemented.

F. Resetting the Extended Kalman Filters

To prevent odometry errors from accumulating continuously in the Kalman Filters, we *reset* all filters to initial states if one of the following situations happens: a) the robot has moved for a certain long distance d_{reset}, since the last perceptual information is received; b) the robot has performed actions for a certain long time t_{reset}, since the last perceptual information is received; c) the offset between the filters' last state and the current perceptual information exceeds d_{error} (This situation usually happens when the robot is moved by external forces, see Sec. V-B). Thresholds d_{reset},t_{reset}, and d_{error} can be determined through experiments in real environments. As we have described in section C, the initial state of a certain landmark will be determined after the "first" detection of it, counting from this resetting time step. Then, a new recursive process can be implemented.

IV. Detailed Perception Model

A. For a Single Landmark

Assume that the landmark's distance and angle relative to the robot obtained from the filters in section III are denoted by d_{mea} and α_{mea} respectively. Take the landmark's angle to the robot for example. Assume that the type of landmark's angle is type *i*. For each sample, given the angle from the landmark to it as α_{spl}, this type (the *i*th type) of sample's *p*-value can be calculated by

$$p_i(\alpha_{mea}, \alpha_{spl}, \sigma_i) = \frac{1}{\sqrt{2\pi}\sigma_i} \exp\left[-\frac{(\alpha_{spl} - \alpha_{mea})^2}{2\sigma_i^2}\right]. \tag{20}$$

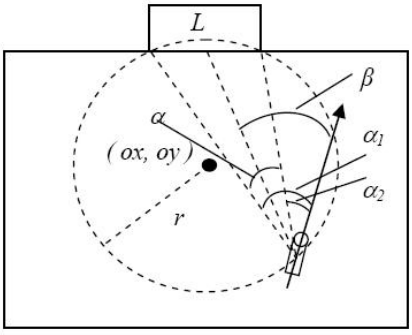

Fig. 2. The circle with two landmarks and the robot on its circumference.

Here we use the Gaussian distribution to describe how α_{mea} influences current sample's *p*-value (and the influence on the final localization result.)

Take the distance for another example. For each sample, given the distance from the landmark to it as d_{spl} , the type (the *j*th type) of sample's *p*-value can be calculated by

$$p_j(d_{mea}, d_{spl}, \sigma_j) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left[-\frac{(d_{spl} - d_{mea})^2}{2\sigma_j^2}\right] \tag{21}$$

σ_i and σ_j in (20) and (21) are constants, affecting to what extent perceptual information of the *i*th and *j*th type can influence the final localization result. They are empirical values determined in experiments in the actual environment. A typical example would be $\sigma_i = 3.16^\circ$ and $\sigma_j = 15.8mm$.

B. For Two Landmarks

When a pair of landmarks is observed, we will combine such perceptual information to form an approach. See Fig. 2 for details. From the landmarks' angles, denoted by α_1 and α_2, we can figure out the angle between them, namely α = $\mid \alpha_1$ - $\alpha_2 \mid$. With a fixed line segment (formed by two landmarks) as the chord, and α as the angle in a circular segment, we can draw two circles. The centre of one of the circles, denoted by $((ox, oy), r_{mea})$, should be within the environment, while that of the other is not. Both landmarks and the robot should be on the former circle. For each sample, given the distance from the centre to it as d_{spl}, the type (the *k*th type) of sample's *p*-value can be calculated by

$$p_k(r_{mea}, d_{spl}, \sigma_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left[-\frac{(d_{spl} - r_{mea})^2}{2\sigma_k^2}\right] \tag{22}$$

where σ_k in (22) is a constant, just as σ_i and σ_j in (20) and (21). A typical example would be $\sigma_k = 14.2mm$.

V. Experiment Setup, Results and Discussion

A. Experiment Environment and Test-bed Robots

RoboCup International Competitions and Conferences [16] is one of the largest robot soccer activities. The annual event was founded in 1997 and attracts more and more researchers ever since. One of the competition leagues, the Legged league, provides especially challenging localization problems.

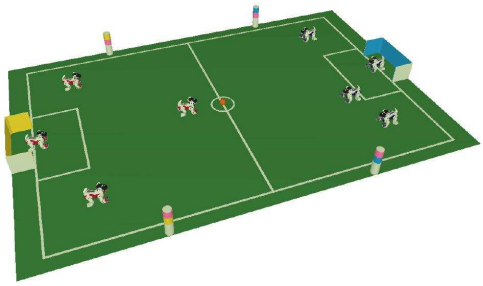

Fig. 3. Competition field of the Legged league. With the landmarks(unique goals and colored markers), a robot can localize itself.

As shown in Fig. 3, robots play soccer on a $5.4m \times 3.6m$ green carpet with two colored goals at each end and four visually dissimilar artificial markers on the outer periphery. Goals and markers serve as the robot's primary visual landmarks for localization.

In our experiment, we use the Sony AIBO ERS-7 robot [14]. All the processing is performed on board using a 576MHz CPU and a 32M memory. It has 20 degrees of freedom: 3 in each leg, 3 in its head and 5 in other parts. The most central sensor is the CMOS color camera located on its head. The camera provides YUV images at 30 frames per second and has a horizontal field-of-view of 56.9° and a vertical field-of-view of 45.2°. Although it has other sensors like touch sensors, accelerometers and distance sensors, these sensors cannot contribute much to self-localization.

B. Uncertainties in this Dynamic Uncertain Environment

The RoboCup Legged league soccer competition holds under a dynamic uncertain environment [15]. Here we will illustrate some of the uncertainties that have great influence on robot self-localization.

1) Uncertainties in the Robot's Odometry Data: Uncertainties in the robot's odometry data can be attributed to a number of factors, including: time lag (the data is not sent until an action has been performed); robots' congestion (robots collide with each other without awareness); slip on the carpet (commonly happens especially when using a different carpet); individual robot motor differences (the same control parameters may result in different obometry data).

2) Uncertainties in Perceptual Information: Sensor readings from the CMOS camera have perceptual errors due to the following reasons:

a) When the robot is walking fast with its head scanning around, objects in images will appear distorted, making it difficult to recognize the objects' edges, resulting in inaccurate and even false perceptual information.

b) A landmark's distance to the robot is in inverse proportion to the amount of pixels that depicted it. Such feature may induce large errors especially when landmarks faraway are perceived. For example, for a landmark corresponding to 4 or 5 pixels in images, a difference of one pixel may lead to a difference of up to around $100cm$ in distance calculation.

c) As indicated in [12], shadows and reflections in different parts of the field will make the colors change and cause them to be misclassified. Different lighting conditions have a significant impact on the accuracy of perceptual information.

3) Uncertainties Due to External Forces: According to the rules of the RoboCup legged league, when a robot commits a foul, it is picked up by the referee and placed at a different point on the field after a period of penalty time. It has to re-localize itself promptly once it gets started again. The robot is kidnapped in this case.

TABLE I

RESULTS FOR STABILITY

	Average Error	Standard Deviation	Maximal Error
Landmark Distance	$35.2cm$	$18.8cm$	$137.0cm$
Landmark angle	$11.1°$	$5.3°$	$29.0°$
Position (MCL)	$12.3cm$	$5.2cm$	$31.1cm$
Position (MCL-EKF)	$4.5cm$	$2.2cm$	$14.9cm$

C. Test on Stability & Reduction of Perceptual Errors

The aim of this test is to show to what extent MCL-EKF can reduce errors in perceptual information and increase the *stability* of localization results, which refers to the ability of a robot to stay localized at the desired position once it has arrived. We conduct experiments on a goalie robot, because the goalie always walks within a rather small penalty area (Fig. 3), thus it is convenient to measure the actual perceptual information of landmarks and to examine the stability. Once the goalie catches sight of a ball, it will calculate a blocking pose, and will stop automatically when the localization result suggests that blocking pose. Once it stops, we record its x, y coordinates (excluding orientation, because the robot is turning around all the time) and the landmark's distance and angle within it's field of vision. We conduct the experiment for five minutes. Results are listed in Table I.

We can conclude from Table I that with EKF enhancement, errors of robot position can be reduced by approximately two thirds comparing with MCL only, indicating a dramatic increase in Stability. In addition, perceptual errors can be reduced by nearly six sevenths through our localization method.

We also find that when the value of perceptual information (say distance or angle of landmarks) vibrates, say around a rather fixed value, the result of MCL also vibrates (and sometimes just "jump" from one position to another) but cannot lead to a final stable position. On the contrary, the result of MCL-EKF vibrates with a smaller swing as time goes on, and finally results in a stable position. This property of MCL-EKF is especially useful when stability counts greatly, e.g. for a goalie robot to occupy a pivotal place *constantly* to block opponents' shoots.

D. Test on Precision

Next we plan to test the precision of localization results when the robot is moving. It is assigned a task to visit a

TABLE II

RESULTS FOR PRECISION

	Average Error	Standard Deviation
Position (MCL)	$15.1cm$	$6.5cm$
Position (MCL-EKF)	$9.2cm$	$4.2cm$

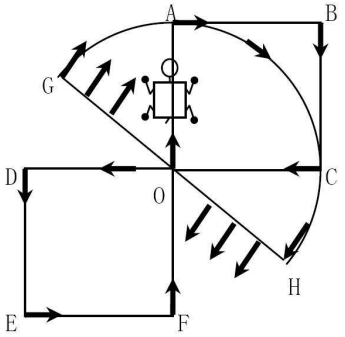

Fig. 4. A complex path designed for test on ability of recovery.

sequence of 10 points during which it is allowed to constantly scan the environment. Without loss of generality, these points are randomly chosen out of a pool of nearly 30 points equally distributed in the field. It will pause 10 seconds indicating that it has reached the desired point, and allows us to record the localization errors in that interval. The process is repeated for 20 times. Table II shows the experiment result. MCL-EKF reduces localization errors by nearly one thirds, providing an increase in precision.

We conduct the experiment with rigorous parameters, i.e., there are no field walls around the field to reduce perceptual misreading, the robot walks at a maximal speed (about $350mm/s$), the head scanning frequency is set to "fast" mode. These settings lead to high impact of uncertainties in the environment on the localization errors. The average error will be smaller under a different (competition) condition.

When we compare test on precision with the previous test on stability, we find that error of MCL increases a small amount ($2.8cm$), while that of MCL-EKF increases more ($4.7cm$). We can expect this error increase because there are more uncertainties in this test than in the previous one. MCL-EKF seems to be disturbed more by these uncertainties. To our understanding, this is because situation a and b in Section III-F happen, thus makes EKFs reset themselves more frequently.

E. Test on Ability of Recovery

Here we want to test the problem of kidnapped robot. The robot is assigned a task to try to walk along a complex path on the field (Fig. 4) whatever happens, with its camera repeatedly scanning the environment. In each run, we manually replace the robot through the following two ways.

1) Blocking Simulation: We obstruct the robot with a piece of chipboard to make it slip and remove the chipboard after 3 seconds to allow it to walk on. This experiment simulates the situation when a robot collides with other robots and cannot execute its action commands correctly. We perform such run for 20 times, and measure the time taken to recover its position. A robot is considered to have *recovered* its position when the difference between its actual position and its localization result is less than half of its body length (approximately 14cm). In this way, we have the average value listed in Table III. We can see that MCL-EKF outperforms MCL in this blocking test.

TABLE III

RESULTS FOR BLOCKING SIMULATION

	Average Frames	Approximate time
MCL	34	$1.13sec$
MCL-EKF	28	$0.93sec$

TABLE IV

RESULTS FOR EXTERNAL MOVEMENT SIMULATION

	Average Frames	Approximate time
MCL	73	$2.43sec$
MCL-EKF	67	$2.23sec$

2) External Movement Simulation: We pick up the robot, and promptly place it at another location about 60,120,180,240 or 320cm away, making it face a random direction (should be in any direction). Then it has to walk on from the new location. This experiment simulates the robot kidnapping situation described in section B-3 when the robot is placed at a different point on the field after committing a foul. The definition of recovery is the same as above. After 30 runs, we have the average value of recovery time listed in Table IV. This time, MCL-EKF costs a little less recovery time.

In both simulations, MCL-EKF does better than MCL only. One reason is that in MCL-EKF, the filters reset at the proper time, to prevent further error accumulation. The other lies in the fact that MCL-EKF reduces perceptual errors, thus with the efficient resampling process, the robot re-localizes more quickly.

F. Visual Description of the Localization Process

Fig. 5 depicts a typical process during our experiments. Fig. 5(a) shows that in the initial state, all samples are uniformly and randomly distributed in the field. In (b), the robot reaches a stable state in a short time after the initial state. (c) shows that in the condition of temporary high-noised perceptual information, the sample set diffuses, resulting in a localization result without big errors. (d)-(f) describes the process of the "kidnapped robot" test. In (d), the robot has been replaced far away from its original position very quickly. In (e), the robot resets all Extended Kalman Filters of the landmarks because situation c of section III-F happens. In this situation, all samples' *p*-values are low, thus the sample set diffuses promptly according to our resampling process. (f) shows that after a few valid frames, some samples have reached the right range that the perceptual information suggests, and the robot obtains a precise and stable localization again. The whole process, from (c) to (f), is within 3 seconds.

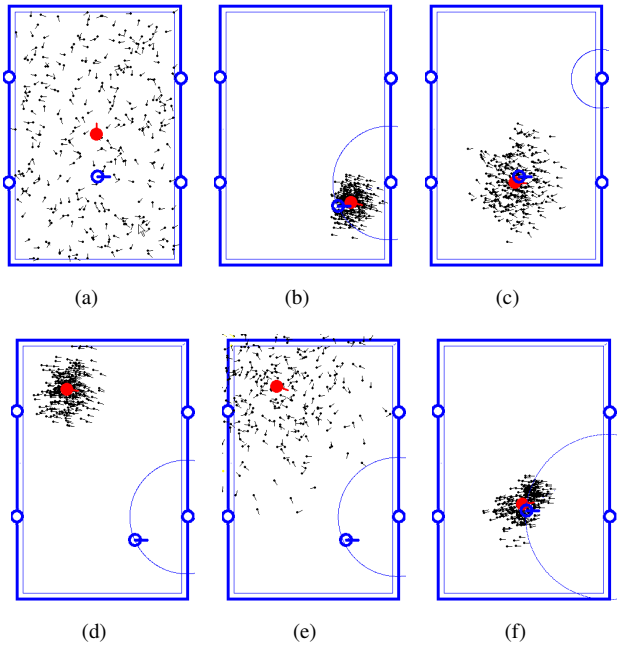

Fig. 5. A typical localization process. Each tadpole-shaped object denotes a sample with the body showing its position and the tail describing its orientation. The biggest tadpole with a hollow core denotes the actual position of the robot, and the other big tadpole with a solid core denotes the localization result.

G. Discussions

Gutmann et al. [6] have conducted experiments to compute the accuracy of color marker detection, concluding that errors in distance and angle can be approximated by Gaussian models. This also happens to errors in goal detection because their detection methods are similar. So establishing EKFs on these landmarks makes sense.

As the number of samples N increases, the performance of the MCL-based systems will be improved only a little. We use $N = 200$ in our experiment, which is enough for competition. The whole MCL-EKF algorithm and the pure MCL algorithm (working on the real robot) average about 3.2ms and 3.0ms for every perceptual information, respectively. This is because MCL has to calculate the p-value for each of the 200 samples while EKF only makes predictions and corrections for at most 6 landmarks. And in comparison, the vision system needs about 18.6ms to provide such perceptual information. So the rather small averaging time of MCL-EKF helps robots to make decisions in real-time.

The MCL-EKF system described above has been applied in RoboCup 2005 4-Legged League Competitions and RoboCup China Open 2005, helping strikers to shoot correctly, the defenders to run to a right supporting position, and the goalie to stay on a favorable blocking position.

VI. Conclusion

In this paper, we present a vision-based localization method called Monte Carlo - Kalman localization (MCL-EKF). This method is a combination of Monte Carlo localization (MCL) and Extended Kalman Filter (EKF) enhancement. We firstly give a detailed implementation of MCL with the emphasize of dealing with multiple types of perceptual information and the problem of robot kidnapping. Next, we establish EKFs on landmarks to build a real-time environment around the robot. Information from this real-time environment will be utilized by the perception model of MCL. We also elaborate on our methods of dealing with a single or two landmarks in the perception model. We conduct different experiments on the test of stability, reduction of perceptual errors, precision, ability of recovery. Results show that the MCL-EKF outperforms MCL, reduces perceptual errors, increases precision and stability and keeps a good ability of recovery. This localization approach will be effective in a known environment, especially those environments which are in need of good precision and stability but are only equipped with high-noised sensors.

Acknowledgment

The authors would like to thank the Wrighteagle team for their efforts in developing the software used as a basis for the work reported in this paper. This work is supported by National Natural & Science Foundation (60275024).

References

[1] J. Leonard and H. Durrant-Whyte, *Mobile robot localization by tracking geometric beacons*. IEEE Transactions on Robotics and Automation 7,3, page 376-82, 1991.

[2] J. Borenstein, H. R.Everett, and L. Feng, *Navigating mobile robots: sensors and techniques*.A. K. Peters, Ltd., Wellesley, MA., 1996.

[3] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo localization: Efficient position estimation for mobile robots. *Proc. of the National Conference on Artificial Intelligence. 1999.*

[4] R. Negenborn. *Robot Localization and Kalman Filters.* MS thesis to institute of information and computing sciences of Utrecht University. Thesis number: INF/SCR-03-09. September 2003.

[5] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. *Markov localization for mobile robots in dynamic environments*. Journal of Artificial Intelligence. 11, 1999.

[6] J. S. Gutmann. Markov-Kalman Localization for Mobile Robots. *In Proceedings of the International Conference on Pattern Recognition (ICPR'02), Quebec, Canada, 2002*

[7] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. *Robust Monte Carlo localization for mobile robots.* Journal of Artificial Intelligence. 2001.

[8] S. Lenser and M. Veloso. Sensor resetting localization for poorly modelled mobile robots. *In The International Conference on Robotics and Automation. April 2000.*

[9] T. Rofer and M. Jungel. Vision-based fast and reactive Monte Carlo localization. *In The IEEE International Conference on Robotics and Automation, pages 856-861, Taipei, Taiwan, 2003.*

[10] M. Sridharan, G. Kuhlmann, and P. Stone. Practical vision-based Monte Carlo localization on a legged robot. *In IEEE International Conference on Robotics and Automation. April 2005.*

[11] J. S. Gutmann , W. Burgard , D. Fox and K. Konolige, An experimental comparison of localization methods. *In Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 1998.*

[12] L. Freeston, R. Middleton. *Applications of the Kalman Filter algorithm to robot localisation and world modeling.* Electrical engineering final year project. University of Newcastle, Australia. 2002.

[13] G. Welch and G. Bishop. *An Introduction to the Kalman Filter.* Department of Computer Science, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-3175. May 23, 2003.

[14] *The Sony Aibo robots.* http://www.aibo.com.

[15] X. Zhang, X. Chen, J. Li, and X. Li. *A Vision-based Monte Carlo Self-Localization System on a Legged Robot.* To appear in Robot, Vol. 28, Issue 4, 2006.

[16] *RoboCup International Competitions and Conferences* http://www.robocup.org