Smooth Variable Structure Filter based SLAM Implementation for a Mobile Robot Navigation

ALLAM Ahmed(1) ,HAMMOUCHE Mounir(2), NEMRA Abdelkrim(3) and HAMERLAIN Mustapha(4)
Laboratory *Robotics*
Department Control Unit
University of Ecole Militaire Polytechnique, Bordj El-Bahri, Algiers
BP 17
ahmedallam900@gmail.com(1), mounirhammouche@hotmail.fr(2), karim_nemra@yahoo.fr(3), hamerlain@cdta.dz(4)

***Abstract* - Simultaneous localization and mapping (SLAM) is vital for autonomous robot navigation. The robot must build a map of its environment while tracking its own motion through that map. There are many ways to approach the problem, mostly based on the sequential probabilistic approach, based around extended Kalman filter (EKF) or the Rao-Blackwellized particle filter. In order to improve the SLAM solution and to overcome some of the (EKF) and (PF) limitations, especially when the process and observation models contain uncertain parameters, we propose to use a robust approach to solve the SLAM problem based on variable structure theory. The new alternative called Smooth Variable Structure Filter (SVSF) is a predictor corrector estimator based on sliding mode control and estimation concepts. It has been demonstrated that the (SVSF) is stable and very robust face modeling uncertainties and noises. Visual SVSF-SLAM is implemented, validated and compared with EKF-SLAM algorithm. The comparison proofs the efficiency and the robustness of localization and mapping using SVSF-SLAM.**

Index Terms - Mobile navigation; filtering; SLAM; EKF; SVSF.

I. INTRODUCTION

The ability of a mobile robot to localize itself is critical to its autonomous operation and navigation. A robot that navigates using maps must be able to accurately localize itself. Consequently, there has been considerable effort on the problem of mobile robot localization and mapping. This problem is known as simultaneous localization and mapping SLAM and there is a vast amount of literature on this topic [1], [2], [3].

SLAM is a technique used by mobile robots to build a map of an unknown environment while simultaneously tracking its own motion. This presents a chicken and egg situation: an accurate map is necessary for localization, and accurate localization is necessary to build a map. The interdependency between the estimates of the robot location and the map of the environment makes SLAM an interesting and difficult research problem. There are many ways to approach the problem, mostly based on the sequential probabilistic approach, based around extended Kalman filter (EKF) [4] or the Rao-Blackwellized particle filter [5].

The use of the classical solution to SLAM based on the Extended Kalman Filter EKF-SLAM, displays several shortcomings such as quadratic complexity and sensitivity to false associations, problems due to the employment of linearized models of nonlinear motion and observation models and so inherits many caveats[2]. Nonlinearity and errors modeling can be a significant problem for EKF-SLAM and leads to inevitable and sometimes dramatic inconsistency in solutions, further the assumption of Gaussian additive noise is often violated, which affects the vehicle and map state estimation and can lead to estimation process divergence. Convergence and consistency can only be guaranteed in the linear case with Gaussian additive noise.

While EKF-SLAM and FastSLAM are the two most popularsolution methods, newer alternatives, which offer much potential, have been proposed, including the use of the unscented Kalman filter (UKF) proposed by Julier and Uhlmann in SLAM [6].Unlike the (EKF), the (UKF) uses a set ofchosen samples to represent the state distribution. The UKF-SLAM avoids the calculation of the Jacobian and Hessian matrices but also obtain higher approximation accuracy with the unscented transformation (UT). However, for high-dimensional systems, the computation load is still heavy; thus, the filter converges slowly. The cubature Kalman filter (CKF) based on a cubature transform which is more accurate and its complexity computation is lower than the one of the (UKF) was used in SLAM [7]. The State Dependent Riccati Equations (SDRE) nonlinear filtering formulation was also used with SLAM which avoids the linearization step [8].

However, the above filters are all based on the framework of the Kalman filter (KF); it can only achieve a good performance under the assumption that the complete and exact information of the process model and the noise distribution which has to be known as a prior. But in practice, the prior noise statistic is usually unknown totally, and the process and

observation models may be not well known or contain modeling uncertainties, thus, the state estimation will have large errors, or, even, the filters will be possible to diverge.

To overcome some of these limitations, we propose to use the (SVSF) filter to solve the SLAM problem. Introduced on 2007, the smoothing variable structure filter (SVSF) is relatively a new filter [9]. It is a predictor correct estimator based on sliding mode control and estimation concepts. In its old form, the (SVSF) is not a classical filter in the sense that it does not have a covariance matrix. Mostly used for state and parameter estimation of dynamic system whether linear or nonlinear, accurate fault detection and estimation [10] and tracking applications [11], it has been demonstrated that the (SVSF) is extremely robust and stable to modeling uncertainties and noise [9], [12].

In this work, we describe our efforts to investigate an alternative to the (EKF) based data fusion. This alternative uses the (SVSF) as a fusion filter for SLAM problem. We propose a smoothing variable structure filter (SVSF) implementation of SLAM algorithm using both robot wheel encoders and stereo vision system. Our motivations behind solving the SLAM problem using the (SVSF) is to overcome some (EKF) limitations and to offer more robustness and stability and less-cost computation to the robot pose and map estimation. We first give an overview of the motion model of our robot and the observation model used and also a description of the EKF-SLAM algorithm is presented. We give a summering of the (SVSF) estimation process, and then we explain how the (SVSF) is used to solve the SLAM problem with the derivation of some equations. We then showcase some simulation results from our implementation applied to theoretical datasets. Finally, we draw some conclusions, some final remarks and perspectives.

II. ROBOT MOTION AND STEREO VISION SENSOR MODELS

A. Process Model

We proceed to model the movement of the robot and the noise associated with it. We assume that the robot is operating in planar environments Fig. 1, whose kinematic state, or pose, is summarized by three variables (1):

$$R = (x_r \quad y_r \quad \theta)^T. \tag{1}$$

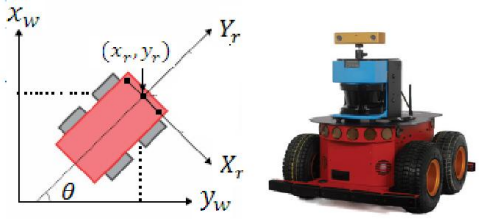

Fig. 1 Robot configuration (Pionner 3-AT).

Where T is the sample period, we assume that the robot is controlled by two velocities: a translational velocity v_k and a rotational velocity w_k at sample k. We note the control input at sample k by $u_k = (v_k\, w_k)^T$. With this control input and the location of the robot at the previous time step we can estimate the robot's current location according to (2)

$$\begin{bmatrix} x_r(k+1) \\ y_r(k+1) \\ \theta(k+1) \end{bmatrix} = \begin{bmatrix} x_r(k) + v_k cos\theta(k)T \\ y_r(k) + v_k sin\theta(k)T \\ \theta(k) + w_k T \end{bmatrix} \tag{2}$$

$$R_{k+1} = f(R_k, u_k). \tag{3}$$

The model (2) states the kinematic for an ideal, noise-free robot. In reality, robot motion is subject to noise, slip or lift. The actual velocities differ from the measured ones by odometer sensor. We will model this difference by a random variable, let:

$$\tilde{u}_k = u_k + n_k = \begin{bmatrix} v_k \\ w_k \end{bmatrix} + \begin{bmatrix} \varepsilon_v \\ \varepsilon_w \end{bmatrix}. \tag{4}$$

The measured velocity equals the true velocity plus some small, additive error (noise) $n_k = (\varepsilon_v\ \varepsilon_w)^T$ [4].

B. Observation Model

In order to execute SLAM the robot needs to be able to choose and track appropriate reference or landmarks in the environment to localize itself. These landmarks must be stable and invariant.

In this work, we opt for a point landmark approach, where the map is a collection of 3D landmark locations. So, how to obtain relative measurement of the landmarks from the images acquired from the stereo vision sensor?

Since we use a stereo vision sensor (two synchronized cameras), the landmarks are as primary visual features (corners, edges…), which must be extracted fromtheacquired images over the time. In the literature several techniques to extract robust features from image, the two most well-known are: the scale invariant feature transform (SIFT) [13] and speeded-up robust features (SURF) [14], then matching operation is done between the visual features extracted in the right and left images. Ones the matching operation is done, the visual features are triangulated to 3D. So the relative measurement delivered by the stereo vision sensor is the 3D coordinates of the landmarks with respect to the robot Fig. 2.

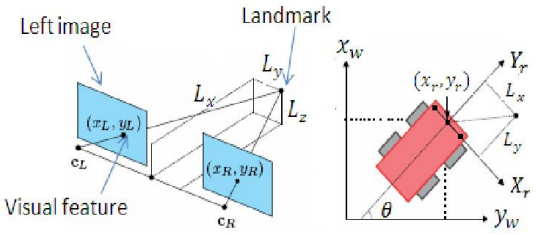

Fig. 2 Stereo vision system geometry.

To validate in simulation the SVSF-SLAM algorithm, we will not use real data, instead of this, theoretical datasets (a set of 3D points) previously generated will be used. When the robot is moving, it detects landmarks (3D points) that are included in the vision sensor field Fig. 3.

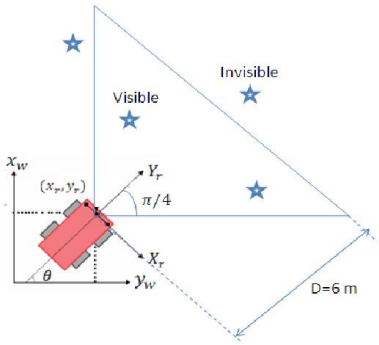

Fig. 3 Sensor vision field.

As said previously, the stereo vision sensor provides relative measurement $Z = (L_x^r \quad L_y^r \quad L_z^r)^T$ of the landmarks with respect to the robot frame, this measurement (observation) will be noted Z.

The model representing the robot frame coordinates of an individual landmark according to its global frame coordinates $L^g = \left(L_x^g \quad L_y^g \quad L_z^g\right)$ and the robot configuration R is called the direct model observation and will be noted:

$$Z = h(R, L^g). \tag{5}$$

$$Z = M_{GR}\begin{pmatrix} L_x^g - x_r \\ L_y^g - y_r \\ L_z^g \end{pmatrix} \tag{6}$$

We denote $M_{GR}$the global to robot projection matrix:

$$M_{GR} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

In reality the observation (measurement) is subject to noise. The real observation model is given in the following model:

$$\tilde{Z} = h(R, L^g) + m_k. \tag{8}$$

Where $m_k = (\varepsilon_x\, \varepsilon_y\, \varepsilon_z)^T$ is the measurement noise.

C. *Inverse Observation Model*

Regarding the dynamic nature of the SLAM algorithm, new observed landmark must be initialized prior to be added to the state vector [6]. The initialization process is in fact the best estimation of the landmark position, and it is a fundamental point to SLAM implementation.

The observation model stated in (6) gives three equations for three dimension variable L^g. The 3D coordinates of a new landmark$\left(L_x^g \quad L_y^g \quad L_z^g\right)$ with respect to the global framework are initialized by solving (5) as follows:

$$L^g = h^{-1}(R, Z). \tag{9}$$

$$L^g = (M_{GR})^{-1}Z + R. \tag{10}$$

III. OVERVIEW OF THE EKF-SLAM ALGORITHM

Our mobile robot is equipped with an odometer sensor providing dead reckoning data, and a stereo vision system providing images of the environment. The odometer sensor may consist of standard wheel encoders attached to the driving wheels of the mobile robot.

The classical solution to SLAM is based on the Extended Kalman Filter EKF-SLAM. It assumes a robot moving in a world with stationary landmarks, the latter being distinctive physical features that can be observed by some sensor (in our case the stereo vision system).

We base our approach to the EKF-SLAM algorithm on the method stated in [4]. Our approach is similar in concept; the main difference is that we changed the way in which the measurements are handled. We found that it is easier to work in Cartesian coordinates when working with images features, while Thrun et al. [4] use polar coordinates.

The (EKF) executes SLAM by including the locations of landmarks in the state vector. Motion and measurement equations are linearized, and motion and measurement noise are approximated with Gaussians, enablingthe use of the normal Kalman filter equations on a non-linear system.

The positions of the landmarks (the map) along with the robot's position at a particular time are considered to be the system state. The problem consists of estimating the new state (robot and landmark positions) at the next time instant given the last movement made by the robot and new observations provided by the stereo vision sensor.

The last movement is usually estimated through odometer (wheel encoders). On the other hand, the stereo vision sensor gives images from the environment. By processing the acquired images, we obtain relative measurement (the 3D coordinates of landmarks with respect to the robot). The estimation process consists of using the (EKF) to fuse the relative measurements and the odometer data in a way that enables robust map-building and localization Fig. 4.

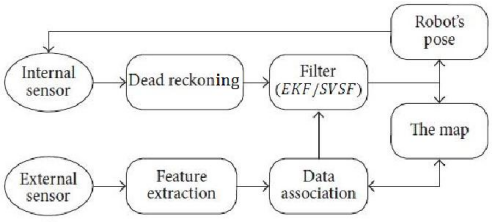

Fig. 4 Sketch of the SLAM algorithm.

The EKF-SLAM algorithm can be divided into two parts: a control update (prediction) where the robot motion is predicted and measurement update using the landmarks observation. As the EKF-SLAM is not the main objective of this work, for more details refer to [3], [4].

IV. THE SMOOTH VARIABLE STRUCTURE FILTER

A new form of predictor-corrector estimator based on sliding mode concepts referred to as the variable structure filter (VSF) was introduced in 2003 [15]. Essentially this method

makes use of the variable structure theory and sliding mode concepts. It uses a switching gain to converge the estimates to within a boundary of the true state values (i.e., existence subspace shown in Fig. 5). In 2007, the smooth variable structure filter (SVSF) was derived which makes use of a simpler and less complex gain calculation [9], [12].

The (SVSF) has been shown to be stable and robust to modeling uncertainties and noise, when given an upper bound on the level of un-modeled dynamics and noise.

The (SVSF) method is model based and may be applied to differentiable linear or nonlinear dynamic equations. The original form of the (SVSF) as presented in [9] did not include covariance derivations. An augmented form ofthe (SVSF) was presented in [16], which includes a full derivation for the filter using covariance matrices.

The basic estimation concept of the (SVSF) is shown in Fig. 5. Some initial values of the estimated states are made based on probability distributions or designer knowledge. An area around the true system state trajectory is defined as the existence subspace. Through the use of the (SVSF) gain, the estimated state will be forced to within this region. Once the value enters the existence subspace, the estimated state is forced into switching along the system state trajectory. A saturation term may be used in this region to reduce the magnitude of chattering or smooth-out the result.

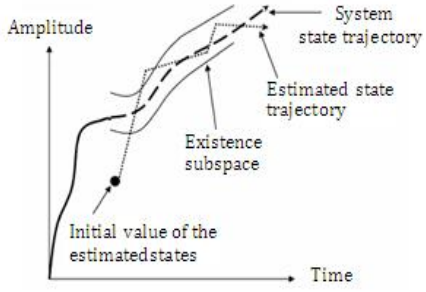

Fig. 5 The SVSFestimation concepts [9].

The estimation process is iterative and may be summarized by the following set of equations (for control or estimation problem)[9], [16]. Like the KF, the system model is used to calculate the a priori state as follows:

$$\hat{X}_{k+1/k} = f(\hat{X}_{k/k}, U_k). \tag{11}$$

Utilizing the predicted state estimates $\hat{X}_{k+1/k}$, the corresponding predicted measurements $Z_{k+1/k}$ and measurement errors $E_{z,k+1/k}$ may be calculated:

$$\hat{Z}_{k+1/k} = h(\hat{X}_{k+1/k}). \tag{12}$$

$$E_{z,k+1/k} = Z_{k+1} - \hat{Z}_{k+1/k}. \tag{13}$$

The (SVSF) process differs from the (KF) in how the gain is formulated. The (SVSF) gain (14) is a function of the a priori and the a posteriori measurement error $E_{z_{k+1/k}}$ and $E_{z_{k/k}}$, the smoothing boundary layer widths φ, the memory or convergence rate matrix γ given by user (with $0 \le \gamma_{ii} \le 1$), as well as the linearzed measurement matrix H.

The (SVSF) guarantees bounded-input bounded-output (BIBO) stability and the convergence of the estimation process by using the Lyapunov stability condition. The derivation of the (SVSF) gain and its stability conditions can be found in [9], [12].The (SVSF) gain matrix is defined as:

$$K_{k+1} = H^{+}\, diag\left[\left(\left|E_{z,k+1/k}\right|_{Abs} + \gamma\left|E_{z,k/k}\right|_{Abs}\right) \circ sat\left(\bar{\varphi}^{-1}E_{z,k+1/k}\right)\right]\left[diag(E_{z,k+1/k})\right]^{-1} \tag{14}$$

Where $\circ$ signifies Schur (or element-by-element) multiplication, the superscript + refers to the pseudo inverse of a matrix and $\bar{\varphi}^{-1}$ is a diagonal matrix constructed from the smoothing boundary layer vector φ, such that:

$$\bar{\varphi}^{-1} = [Diag(\varphi)]^{-1} = \begin{bmatrix} \frac{1}{\varphi_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{1}{\varphi_m} \end{bmatrix} \tag{15}$$

Note that m is the number of measurements, the saturation function of (14) is defined by:

$$sat\left(\bar{\varphi}^{-1}E_{z,k+1/k}\right) = \begin{cases} 1, & E_{zi,k+1/k}/\varphi_i \ge 1 \\ E_{zi,k+1/k}/\varphi_i, & -1 < E_{zi,k+1/k}/\varphi_i < 1 \\ -1, & E_{zi,k+1/k}/\varphi_i \le -1 \end{cases} \tag{16}$$

The gain is used to calculate the update state estimates $X_{k+1/k+1}$:

$$\hat{X}_{k+1/k+1} = \hat{X}_{k+1/k} + K_{k+1}E_{z,k+1/k}. \tag{17}$$

Finally, the update measurement estimate $Z_{k+1/k+1}$ and measurement errors $E_{z,k+1/k+1}$ are calculated, and are used in later iterations:

$$Z_{k+1/k+1} = h(\hat{X}_{k+1/k+1}). \tag{18}$$

$$E_{z,k+1/k+1} = Z_{k+1} - \hat{Z}_{k+1/k+1}. \tag{19}$$

Note that the process estimation needs to be initialized such:

$$\begin{aligned} X_{0/0} &= X_0 \\ E_{z,0/0} &= E_{z,0} \end{aligned} \tag{20}$$

Two critical variables in this process are the a priori and a posteriori measurements (output) error estimates, defined by (13) and (19). Note that (19) is the a posteriori measurement error estimates from the previous time step, and is used only in the gain calculation.

The selection of the smoothing boundary layer width vector φ reflects the level of uncertainties in the filter and the disturbances (i.e., system and measurement noise, and uncertain parameters).

V. THE SVSF-SLAM ALGORITHM

The SVSF-SLAM algorithm applies the (SVSF) to solve the SLAM problem. In doing, the (SVSF) differs from the (EKF), since the (SVSF) is not subject to the Gaussian noise

assumption for the robot motion and the observation model as the (EKF) and doesn't use covariance matrices.

In addition to estimating the robot pose R, the SVSF-SLAM algorithm estimates the coordinates of the landmarks observed along the way. For convenience, we will call the state vector comprising robot pose and the map the combined state vector, and denote this vector y_k. The combined vector is given by $y_k = \left(R_k \underline{L}^{g,k}\right)^T$. Where $R_k = (x_r^k \quad y_r^k \quad \theta^k)^T$ is the robot pose at time k, $\underline{L}^{g,k} = (L_1^{g,k} L_2^{g,k} \dots L_N^{g,k})^T$ is the map vector containing the entire landmarks coordinate at time k, N is the number of the current landmarks, and $L_i^{g,k} = (L_{i,x}^{g,k} L_{i,y}^{g,k} L_{i,z}^{g,k})^T$ is the i_{th} landmark coordinates in the map vector at time k.

As the robot moves, the state vector changes according to the standard noise-free kinematic model (2), (3). In SLAM this motion model is extended to the augmented state vector:

$$y_{k+1} = g(y_k, u_k) = \begin{pmatrix} f(R_k, u_k) \\ \underline{L}^{g,k} \end{pmatrix} \tag{21}$$

Only the first three elements are updated with control input. Landmarks supposed fix, remain where they are.

In our approach, we obtain at each sample k a set of m observations denoted by $\underline{Z}_k = (z_{k,1}, z_{k,2}, \dots, z_{k,m})^T$, where each observation $z_{k,j}$ consists of the robot frame coordinates of the observed landmark.

The observation model of SVSF-SLAM algorithm is deduced from (5):

$$z_{k,j} = h(y_k) = h\left(y_{k,i}\right) = h(R_k, L_i^{g,k}). \tag{22}$$

Where j is the index of an individual landmark observation in $\underline{Z}_k$, and i is the index of the observed landmark at time k.

Even the (SVSF) process requires linear observation model, the non-linear observation function h is approximated using a first degree Taylor expansion [12]:

$$h\left(y_{k,i}\right) = h\left(\overline{y}_{k,i}\right) + H_k^j (y_{k,i} - \overline{y}_{k,i}). \tag{23}$$

Here H_k^j is the derivative of h with respect to the full state vectory_k. Since h depends only on two elements of state vector, the robot pose R_k and the location of the i_{th} landmark $L_i^{g,k}$, we can write H_k^jas follows:

$$H_k^j = h_k^j F_{y,i}. \tag{24}$$

Here h_k^j is the Jacobian of h at $\overline{y}_k$, the mean of y_k, calculated with respect to the state variables R_k and $L_i^{g,k}$.

$$F_{y,i} = \begin{bmatrix} 1 & 0 & 0 & 0 \dots 0 & 0 & 0 & 0 & 0 \dots 0 \\ 0 & 1 & 0 & 0 \dots 0 & 0 & 0 & 0 & 0 \dots 0 \\ 0 & 0 & 1 & 0 \dots 0 & 0 & 0 & 0 & 0 \dots 0 \\ 0 & 0 & 0 & 0 \dots 0 & 1 & 0 & 0 & 0 \dots 0 \\ 0 & 0 & 0 & 0 \dots 0 & 0 & 1 & 0 & 0 \dots 0 \\ 0 & 0 & 0 & \underbrace{0 \dots 0}_{3i-3} & 0 & 0 & 1 & \underbrace{0 \dots 0}_{3N-3i} \end{bmatrix} \tag{25}$$

The execution procedure of the SVSF-SLAM algorithm can be divided into three major steps:

A) Initialization

In SLAM, the initial pose is taken to the origin of the coordinate system. Using the inverse observation model (10) we can initialize thefirst landmarks. As a result we obtain the first state vector $\overline{y}_0$.The SVSF-SLAM algorithm requires also the initialization of the a posteriori measurements (output) error vector $\underline{E}z_0$.

$$\overline{y}_0 = \left(\bar{R}_0 \quad \bar{L}_1^{g,0} \quad \bar{L}_2^{g,0} \quad , \dots, \quad \bar{L}_{N_0}^{g,0}\right)^T.$$
$$\underline{E}z_0 = \left(Ez_0^1 \quad Ez_0^2 \quad , \dots, \quad Ez_0^{N_0}\right)^T.$$

Where $\bar{R}_0 = (0\ 0\ 0)^T$ and N_0 is the number of initial landmarks.

B) Control update

1. $\hat{y}_{k+1} = g(\overline{y}_k, \tilde{u}_k)$.

C) Measurement update

2. **For** all landmarks observations $z_{k+1,j}$.
3. $\hat{z}_{k+1,i} = h(\hat{y}_{k+1}) = h(\hat{R}_{k+1}, L_i^{g,k+1})$.
4. **If** landmark ***j*** is seen before (correspondence found)
5. $\hat{E}z_{k+1} = z_{k+1,j} - \hat{z}_{k+1,i}$.
6. $(H_{k+1}^j)^+ = (F_{y,i})^T (h_{k+1}^j)^+$.
7. $K_{k+1} = (H_{k+1}^j)^+ diag\left[\left(\left|\hat{E}z_{k+1}\right|_{abs} + \gamma\left|Ez_k^i\right|_{abs}\right) \circ sat\left(\bar{\varphi}^{-1}\hat{E}z_{k+1}\right)\right]\left[diag(\hat{E}z_{k+1})\right]^{-1}$
8. $\hat{y}_{k+1} = \hat{y}_{k+1} + K_{k+1}\hat{E}z_{k+1,i}$.
9. $Ez_{k+1}^i = z_{k+1,j} - h(\hat{R}_{k+1}, L_i^{g,k+1})$.
10. **End If**
11. Use (10) to initialize the new landmark$L_{New}^{g,k}$.
12. Add $L_{New}^{g,k}$ to the system state vector $\hat{y}_{k+1}$.
13. Use (19) to initialize the a posteriori measurement error of the new landmark Ez_{k+1}^{New}.
14. Add Ez_{k+1}^{New} to the a posteriori measurement error vector $\underline{E}z_{k+1}$.
15. $N = N + 1\%$ the total number of current landmarks.
16. **End For**
17. $\overline{y}_{k+1} = \hat{y}_{k+1}$.
18. **Return** $\overline{y}_{k+1}, \underline{E}z_{k+1}$.

Where j is the index of an individual observation in $\underline{Z}_{k+1}$ and i is the index of its associated landmark in the system state vector.

Note that only the observation model is linearized, there is no need to linearize the motion model as in (EKF) estimation process. Another important remark in the SVSF-SLAM estimation process is that, for each observation $Z_{k+1,j}$, only the

robot pose R_{k+1} and the observed landmark $L_i^{g,k+1}$ will be updated, while in EKF-SLAM we update the entire system state vector y_{k+1}.

The width of the smoothing boundary layer vector φ can be a function of the upper bound of uncertainties-modeling and noise, the process estimation accuracy depends on the value of φ used [9], [12].

In our work, motion model will be supposed well known, while the observation model will be supposed containing some modeling errors Δh, so the vector φ will depend on the priori knowledge of the upper level of motion and observation noise $|n_k|_{Max}$,$|m_k|_{Max}$ and $|\Delta h|_{Max}$[9].

VI. SIMULATION

A. Simulation Environment

We created the environment with the aim of simulating thereal world without it being unnecessarily complicated. We opted for a circular trajectory with landmarks on the robot way. At each time step an input command is generated and stored to be used in the SLAM simulations but, before the robot executes the command, we add to it some noise to simulate the uncertainty.

As the robot moves through the environment, landmarks in the robot's field of view are included in the measurement Z at every time step. Then we add noise to the measurements coordinates vector Z.

In order to verify the efficiency of the SVSF-SLAM comparing with the EKF-SLAM, some different simulations were done with different environment noise. Suppose that the observation noise obeys the mixture Gauss distribution as $n_k \sim 0.5N(0,R_1) + 0.5N(0,R_2)$.

$$R_1 = \begin{bmatrix} (0.1)^2 & 0.001 & 0.002 \\ 0.001 & (0.1)^2 & 0.001 \\ 0.002 & 0.001 & (0.1)^2 \end{bmatrix}, R_2 = \begin{bmatrix} 0.002 & 0.01 & 0.02 \\ 0.005 & 0.002 & 0.01 \\ 0.01 & 0.005 & 0.002 \end{bmatrix} \quad (26)$$

And the motion noise obeys to Gauss distribution as $m_k \sim N(0,Q)$.

$$Q = \begin{bmatrix} 0.02 & 0.001 \\ 0.001 & 0.01 \end{bmatrix} \quad (27)$$

The width of the smoothing boundary layer vector φ used is:

$$\varphi = (21 \quad 21 \quad 15)^T \quad (28)$$

B. Simulation Results

To illustrate the solution of the SVSF-SLAM comparing with the EKF-SLAM solution, we present the followings results. First, consider Fig.6 and Fig. 7, which show the result of EKF-SLAM and SVSF-SLAM respectively after the robot has traveled along the real trajectory.

In the both case, the SLAM algorithm builds a map consisting of landmarks, which are marked as circles/squares in the figures. The red path (odometry only) is obviously incorrect. On the other hand the SVSF-SLAM corrected path is consistently following the reference path butter than that of the EKF-SLAM Fig. 8. From figures Fig. 9 and Fig. 10, we learn that the root mean square error (RMSE) values of the map and the robot position estimation of the SVSF-SLAM algorithm is smaller than that of the EKF-SLAM until sample 580, in which we see in the case of EKF-SLAM that there is a sudden reduction of the RMSE values compared with the SVSF-SLAM because the robot has observed landmarks already seen (the loop closing appearance); so we can note that the SVSF-SLAM algorithm accuracy is better than that of the EKF-SLAM. We can also see that the robot pose and the map estimation RMSE in the case of the SVSF-SLAM are nearly limited.

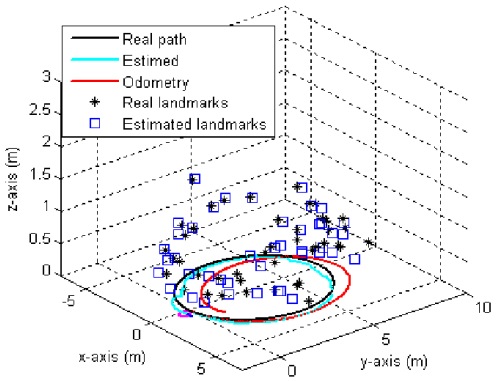

Fig. 6 EKF-SLAM simulation results.

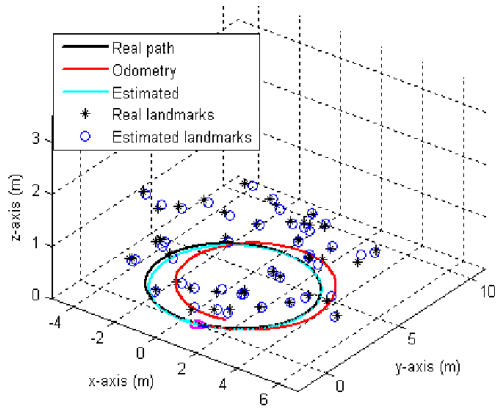

Fig. 7 SVSF-SLAM simulation results.

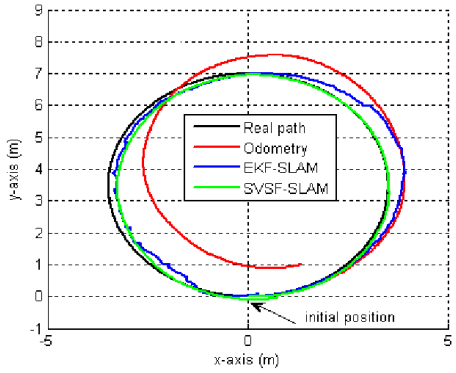

Fig. 8 Comparison of the simulation results with EKF-SLAM and SVSF-SLAM.

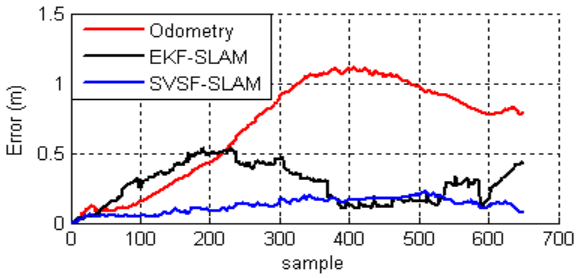

Fig. 9 Robot pose estimation RMSE.

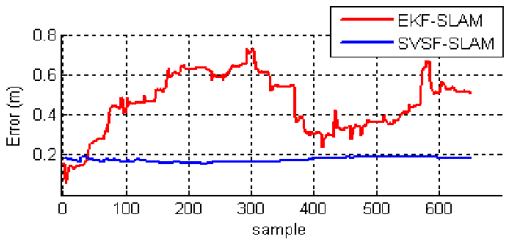

Fig. 10 Map estimation RMSE.

Finally, consider Fig. 11 which shows the increasing of computation time of the EKF-SLAM which is quadratic in N, the number of the current landmarks in the map. In the other hand the SVSF-SLAM time computation is limited and nearly constant, since the algorithm update for each measurement only the robot state and the correspond landmark position.

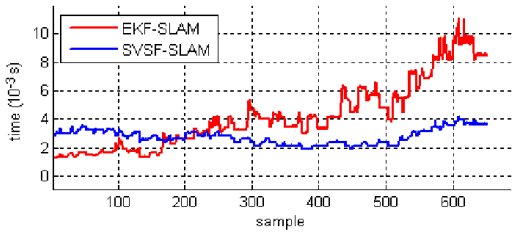

Fig. 11 Comptuation time.

VII. CONCLUSION

In this work, we have developed and implemented a smoothing variable structure filter SLAM algorithm, which is based on 3D observation and odometer. The novel SLAM algorithm based on the (SVSF) estimator doesn't require covariance matrices. It was proposed in order to solve the problem of the unknown noise statistic characteristics of the system in real world and to deal with the parameters uncertainties and linearization errors. The SVSF-SLAM algorithm enables less complex correction gain and low-cost computation and robust navigation given the upper bound of the uncertainty-modeling and noise. It allows more stability and accuracy estimation comparing with EKF-SLAM algorithm.

As future work, we expect to use the (SVSF) filter in its new form using matrices covariance to evaluate the uncertainty of the estimation and also using the (SVSF) with optimal (adaptive) smoothing boundary layer vector to solve the SLAM problem.

REFERENCES

[1] G. Dissanayake, P. Newman, H.F. Durrant-Whyte, S. Clark, and M. Csobra, "A solution to the simultaneous localisation and mapping (SLAM) problem," IEEE Trans. Robot. Automat., vol. 17, no. 3, pp. 229–241, 2001.

[2] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping (SLAM): Part I," IEEE Robotics and Automation Magazine, vol. 13, no. 2, pp. 99–110, 2006.

[3] W. Brink, C. van Daalen, and W. Brink, "Stereo vision as a sensor for EKF SLAM," 22nd Annual Symposium of the Pattern Recognition Association of South Africa, pp. 19–24, 2011.

[4] S. Thurn, W. Burgard, and D. Fox. "Probabilistic robotics," MIT Press 2006.

[5] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," Proceedings of the AAAI National Conference on Artificial Intelligence, 2002.

[6] X. Yan, C. Zhao, and J. Xiao, "A novel fastslam algorithm based on iterated unscented kalman filter," in Proceedings of the IEEEInternational Conference on Robotics and Biomimetics (ROBIO'11), pp. 1906–1911, 2011.

[7] K. P. B. Chandra, D. W. Gu, and I. Postlethwaite, "Cubature kalman filter based localization and mapping," World Congress, vol. 18, no. 1, pp. 2121–2125, 2011.

[8] A. Nemra and N. Aouf, "Robust INS/GPS Sensor Fusion for UAV Localization Using SDRE Nonlinear Filtering," IEEE SENSORS JOURNAL, VOL. 10, NO. 4, april 2010.

[9] S. R. Habibi, "The Smooth Variable Structure Filter," Proceedings of the IEEE, vol. 95, no. 5, pp. 1026-1059, 2007.

[10] S. R. Habibi and R. Burton, "Parameter Identification for a High Performance Hydrostatic Actuation System using the Variable Structure Filter Concept," ASME Journal of Dynamic Systems, Measurement, and Control, 2007.

[11] S. A. Gadsden, D. Dunneb, S. R. Habibia, and T. Kirubarajanb " Comparison of Extended and Unscented Kalman, Particle, and Smooth Variable Structure Filters on a Bearing-Only Target Tracking Problem," Proc. of SPIE Vol. 7445 74450B-13, 2009.

[12] Gadsden, S.A., Smooth Variable Structure Filtering: Theory and Applications, Ph.D. Thesis, Department of Mechanical Engineering, McMaster University, Hamilton, Ontario, Canada, 2011.

[13] D. Lowe, "Object recognition from local scale invariant features," IEEE International Conference on Computer Vision, pp. 1150–1157, 1999.

[14] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool, "Speeded-up robust features (SURF)," Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346–359, 2008.

[15] S. R. Habibi and R. Burton, "The Variable Structure Filter," Journalof Dynamic Systems, Measurement, and Control (ASME), vol. 125,pp. 287-293, September 2003.

[16] S. A. Gadsden and S. R. Habibi, "A New Form of the Smooth Variable Structure Filter with a Covariance Derivation," in IEEE Conference on Decision and Control, Atlanta, Georgia, 2010.