KALMAN FILTERING FOR DIFFERENTIAL DRIVE ROBOTS TRACKING

Luis Aguiar*, Marcos R. O. A. Maximo*, Takashi Yoneyama†, Samuel Pinto*

*Autonomous Computational Systems Lab (LAB-SCA),
Computer Science Division, Aeronautics Institute of Technology
Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900
São José dos Campos, SP, Brazil

†Electronic Engineering Division, Aeronautics Institute of Technology
Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900
São José dos Campos, SP, Brazil

Emails: luis.guilherme@aluno.ita.br, mmaximo@ita.br, takashi@ita.br, sacepi@gmail.com

Abstract— This paper presents the application of the stochastic filtering techniques known as Extended Kalman Filter and Uscented Kalman Filter in the tracking and state estimation of differential drive robots. This work presents and describes these algorithms, the stochastic modeling for this specific problem, the results from implementation using real robot's data and performance evaluation from simulations.

Keywords— Mobile Robots, Stochastic Filtering, Tracking

1 Introduction

Mobile robotics is a rich and growing field in modern technology. Recently, it has been achieving huge technical advancements and at the same time finding several interesting applications from military to domestic scenarios. A key element in the robotics field is the movement tracking and the state estimation of robot agents throughout the environment. As can be seen in some recent sophisticated robots, these tasks play an essential role in the development of advanced autonomous machines.

One particular class of mobile robots is the differential drive robots, which can be widely seen in several applications due to its hardware simplicity and versatility. In this way, an interesting challenge involving these robots is the IEEE Very Small Size competition. VSS is a soccer based challenge with autonomous robots with maximum dimensions of $7.5cm$x$7.5cm$x$7.5cm$. Each team has three robots in the field and is allowed to have a camera fixed above it and also develop a vision system. Most of the processing occur in a external computer which communicates with the agents using wireless interface.

This work is based on our recent efforts for the development of a tracking system for the ITAndroids' VSS team. ITAndroids is a robotics research group at Technological Institute of Aeronautics which was refounded in mid-2011 after some few years of inactivity. The group participates in several national and international competitions. In particular, ITAndroids' VSS team has participated in the 2014 and 2015 editions of the Latin American Robotics Competition (LARC) and Brazilian Robotics Competition (CBR), which happen as one single event, achieving the 4th place in the latest edition.

Tracking is a particularly important task concerning not only the VSS challenge but many other applications. Finding the motion of an object can be useful in navigation, video surveillance, satellite control, missile guidance etc. We have experienced a great progress in the probabilistic approach ()(S. Thrun and Fox, 2005) and (Brookner, 1998)) using Bayesian filtering to deal with tracking. Many problems can be cast into the problem of estimating the state of a dynamic system from discrete noisy measurements. Therefore, the standard formulation includes the state space and discrete-time approach (Arulampalam et al., 2001), using the world famous non-linear Gaussian filters (EKF and UKF) and the non-parametric Particle Filter. In addition to the position estimation, in several situations, it is also necessary to extract the velocity estimates from the measurement of the position alone. For example, one can mention some feedback control models and trajectory planning algorithms which demand an optimal velocity estimation. Some formulations of these bayesian filters aims to solve this issue, as can be seen in (Palumbo et al., 2010) and in (Bruno and Pavlov, 2004).

Therefore, the purpose of this paper is to present the application of the Extended Kalman Filter, Uscented Kalman Filter and the linear Kalman filter in the described VSS tracking problem, as well as assessing its performance based on the Root Mean Square Error (RMSE) calculated for several simulations and also applying it to real data. In this work, Section 1 gives a presentation and an overview of the subject, Section 2 describes the mentioned techniques introducing the correspondent algorithms and Section 3 provides the stochastic model of the differential drive robots in the tracking problem. Section 4 describes the procedure used in the covariance calibration process. Section 5 presents some implementation results using the vision data from a real robot and also some simulation tests. Finally, in Section 6 we can see the main conclusions and final considerations.

2 Probabilistic Localization

In the state-space and discrete time approach to the tracking problem, we represent the robot's pose at time $k \in \mathbb{Z}$ by its state vector $\mathbf{x}_k \in \mathbb{R}^n$. From a probabilistic formulation, we define its state as a vector of random variables, and for representing its state uncertainty we are interested in the probability density function (pdf) of $\mathbf{x}_k$.

In order to use the concept of bayesian filtering, two models are needed (Arulampalam et al., 2001). A model describing the dynamic evolution of the state, called the movement model, represented in equation (1), and a second model relating the noisy measurement to the state, called the measurement model, represented in equation (2).

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \tag{1}$$

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k) \tag{2}$$

In equation (1), f is the propagation function, $\mathbf{u}_k$ is the vector of control commands and $\mathbf{w}_k$ is the process associated random perturbations. Meanwhile, in equation (2), h is the function that transforms the state space into the observation space, with an associated random noise $\mathbf{v}_k$. We shall consider that $\mathbf{w}_k$ and $\mathbf{v}_k$ are independent and have null mean and covariance matrices $\mathbf{Q}_k$ and $\mathbf{R}_k$, respectively. The estimate of $\mathbf{x}_k$ will be denoted by $\hat{\mathbf{x}}_{k|k}$ and its covariance matrix by $\mathbf{P}_{k|k}$.

Given these models, the Bayesian approach aims to construct the posterior state distribution function $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ given all the current measurements up to k. In the next subsections, brief reviews of two famous techniques widely used in the attempt to construct such distribution are presented. They are both non-linear extensions of the well known Kalman Filter, which provides the optimal estimator in the case in which the equations (1) and (2) are linear and the noises and the initial distribution are Gaussian and mutually independent.

2.1 Extended Kalman Filter

The Extended Kalman Filter (EKF) extends the original Kalman Filter by linearizing the models given by equations (1) and (2) around the current estimate. The extended Kalman Filter, despite the non-linearities, still assumes that the state probability function is Gaussian, even though that is not true in general for non-linear functions. The Extended Kalman Filter, in contrast with the linear one, may not be optimal and may produce undesirable results for highly non-linear functions.

For smooth non-linear functions, in which the linearization gives a good local approximation of the function, the Extended Kalman Filter usually has good performance. Equations (3) to (6) show precisely how the linearization is performed. The complete EKF algorithm is presented in Algorithm 1.

$$\mathbf{F}_k = \frac{\partial f_k(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{w})}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}_{k-1|k-1}}^{\mathbf{w}=0} \tag{3}$$

$$\mathbf{L}_k = \frac{\partial f_k(\mathbf{x}, \mathbf{u}_{k-1}, \mathbf{w})}{\partial \mathbf{w}}\Big|_{\mathbf{x}=\mathbf{x}_{k-1|k-1}}^{\mathbf{w}=0} \tag{4}$$

$$\mathbf{H}_k = \frac{\partial h_k(\mathbf{x}, \mathbf{v})}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}_{k|k-1}}^{\mathbf{v}=0} \tag{5}$$

$$\mathbf{M}_k = \frac{\partial h_k(\mathbf{x}, \mathbf{v})}{\partial \mathbf{v}}\Big|_{\mathbf{x}=\mathbf{x}_{k|k-1}}^{\mathbf{v}=0} \tag{6}$$

Algorithm 1 EKF

Initialization
$\hat{\mathbf{x}}_{0|0} = \hat{\mathbf{x}}_0$
$\mathbf{P}_{0|0} = \mathbf{P}_0$

for $k \leftarrow 1$ **to N do**

Prediction Step
$\hat{\mathbf{x}}_{k|k-1} \leftarrow f_k(\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_{k-1}, \mathbf{0})$
$\mathbf{F}_k \leftarrow \frac{\partial f_k}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}}$
$\mathbf{L}_k \leftarrow \frac{\partial f_k}{\partial \mathbf{w}}|_{\mathbf{x}=\hat{\mathbf{x}}_{k-1|k-1}}$
$\mathbf{P}_{k|k-1} \leftarrow \mathbf{F}_k \mathbf{P}_{k-1|k-1} \mathbf{F}_k^T + \mathbf{L}_k \mathbf{Q}_k \mathbf{L}_k^T$

Filtering Step
$\mathbf{H}_k \leftarrow \frac{\partial h_k}{\partial \mathbf{x}}|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}$
$\mathbf{M}_k \leftarrow \frac{\partial h_k}{\partial \mathbf{v}}|_{\mathbf{x}=\hat{\mathbf{x}}_{k|k-1}}$
$\mathbf{S}_k \leftarrow \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{M}_k \mathbf{R}_k \mathbf{L}_k^T$
$\mathbf{K}_k \leftarrow \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1}$
$\hat{\mathbf{x}}_{k|k} \leftarrow \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}))$
$\mathbf{P}_{k|k} \leftarrow (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}$

end for

2.2 Unscented Kalman Filter

The EKF may perform poorly when the prediction and observation functions are highly nonlinear. The Unscented Kalman Filter (UKF) uses a deterministic sampling technique, known as the unscented transform (UT). This method better approximates the posterior's true mean and covariance (Julier and Uhlmann, 2004). Therefore, the UKF often provides better estimate than the EKF with similar computational cost. Additionally, the UKF does not require Jacobians computations, which may be a hard task for very complex models. The UT carefully picks samples (called sigma points) around the mean and propagate them through the nonlinear function, from which the mean and covariance of the estimate are computed. Algorithm 2 shows a pseudocode for the UKF, where the sigma points are selected using the function DrawSigmaPoints, presented in Algorithm 3.

In the function DrawSigmaPoints, α and β are constants, where α determines the spread of the sigma points around the mean and is usually set to a small value (we use $\alpha = 10^{-3}$) and $\beta = 2$ is the optimal value for gaussian distributions. Moreover, $\lambda = \alpha^2(n+\kappa)$, where κ is an auxiliary constant usually set to 0. The notation $\sqrt{\mathbf{P}}$ means the square-root matrix of $\mathbf{P}$, i.e. the matrix $\mathbf{F}$ such as $\mathbf{F}\mathbf{F}^T = \mathbf{P}$. The square-root of a matrix may be computed using the Cholesky decomposition. Moreover, $\mathbf{F}_{:,i}$ is the i-th column of the matrix $\mathbf{F}$.

Algorithm 2 Unscented Kalman Filter

Initialization
$\hat{\mathbf{x}}_{0|0} = \hat{\mathbf{x}}_0$
$\mathbf{P}_{0|0} = \mathbf{P}_0$

for $k = 1, \ldots, N$ **do**
  Prediction
  $\mathbf{x}^a = \begin{bmatrix}\hat{\mathbf{x}}^T_{k-1|k-1} & \mathbf{0}^T_m\end{bmatrix}^T$
  $\mathbf{P}^a = \mathrm{diag}(\mathbf{P}_{k-1|k-1}, \mathbf{M}_k)$
  $[\boldsymbol{\chi}, \omega_m, \omega_c] = \mathrm{DrawSigmaPoints}(\mathbf{x}^a, \mathbf{P}^a)$
  for $i = 0, \ldots, 2n$ **do**
    $\boldsymbol{\chi}^{(i)}_{k|k-1} = f_k(\boldsymbol{\chi}^{(i)}_x, \boldsymbol{\varepsilon}_{k-1}, \boldsymbol{\chi}^{(i)}_w)$
  end for
  $\hat{\mathbf{x}}_{k|k-1} = \sum_{i=0}^{2n} w^{(i)}_m \boldsymbol{\chi}^{(i)}_{k|k-1}$
  $\mathbf{P}_{k|k-1} = \sum_{i=0}^{2n}\left\{w^{(i)}_c\left(\boldsymbol{\chi}^{(i)}_{k|k-1} - \mathbf{x}_{k|k-1}\right) \times \left(\boldsymbol{\chi}^{(i)}_{k|k-1} - \mathbf{x}_{k|k-1}\right)^T\right\}$

  Filtering
  $\mathbf{x}^a = \begin{bmatrix}\hat{\mathbf{x}}^T_{k|k-1} & \mathbf{0}^T_r\end{bmatrix}^T$
  $\mathbf{P}^a = \mathrm{diag}(\mathbf{P}_{k|k-1}, \mathbf{R}_k)$
  $[\boldsymbol{\chi}, \omega_m, \omega_c] = \mathrm{DrawSigmaPoints}(\mathbf{x}^a, \mathbf{P}^a)$
  for $i = 0, \ldots, 2n$ **do**
    $\gamma^{(i)}_k = h_k(\boldsymbol{\chi}^{(i)}_x, \boldsymbol{\chi}^{(i)}_v)$
  end for
  $\hat{\mathbf{z}}_k = \sum_{i=0}^{2n} w^{(i)}_m \boldsymbol{\gamma}^{(i)}$
  $\mathbf{P}_{zz} = \sum_{i=0}^{2n} w^{(i)}_c \left(\boldsymbol{\gamma}^{(i)}_k - \hat{\mathbf{z}}_k\right)\left(\boldsymbol{\gamma}^{(i)}_k - \hat{\mathbf{z}}_k\right)^T$
  $\mathbf{P}_{xz} = \sum_{i=0}^{2n}\left\{w^{(i)}_c\left(\boldsymbol{\chi}^{(i)}_{k|k-1} - \mathbf{x}^a\right) \times \left(\boldsymbol{\gamma}^{(i)}_k - \hat{\mathbf{z}}_k\right)^T\right\}$
  $\mathbf{K}_k = \mathbf{P}_{xz}\mathbf{P}^{-1}_{zz}$
  $\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{z}_k - \hat{\mathbf{z}}_k)$
  $\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{P}_{zz}\mathbf{K}^T_k$
end for

Algorithm 3 Function DrawSigmaPoints.

function DrawSigmaPoints($\mathbf{x}^a$, $\mathbf{P}^a$)
  $\boldsymbol{\chi}^{(0)} = \mathbf{x}^a$
  $w^{(0)}_m = \frac{\lambda}{n+\lambda}$
  $w^{(0)}_c = \frac{\lambda}{n+\lambda} + (1 - \alpha^2 + \beta)$
  for $i = 1, \ldots, n$ **do**
    $\boldsymbol{\chi}^{(i)} = \mathbf{x}^a + \left(\sqrt{(n+\lambda)\,\mathbf{P}^a}\right)_{:,i}$
    $w^{(i)}_m = w^{(i)}_c = \frac{1}{2(n+\lambda)}$
  end for
  for $i = n+1, \ldots, 2n$ **do**
    $\boldsymbol{\chi}^{(i)} = \mathbf{x}^a - \left(\sqrt{(n+\lambda)\,\mathbf{P}^a}\right)_{:,i-n}$
    $w^{(i)}_m = w^{(i)}_c = \frac{1}{2(n+\lambda)}$
  end for
end function

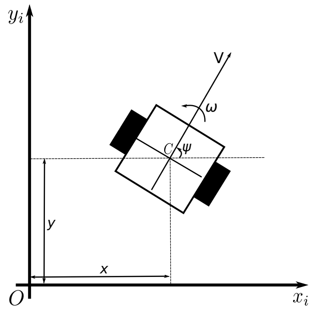

Figure 1: Diferential Drive Robot Geometry

3 Stochastic Modeling

In simple terms, differential drive robots constitute a class of robots which only have two active wheels. Figure 1 illustrates its geometrical and dynamical parameters. For tracking, we shall work with two stochastic models for a differential drive robot, each one with a different state representation.

In the first model, called Unicycle model, our state is given by $\mathbf{x}_k = [x_k\ y_k\ \psi_k\ v_k\ \omega_k]^T$ in which x_k, y_k and ψ_k are the coordinates and orientation of a fixed point between the robot's wheels according to a reference frame. In the tracking problem of the opponent's robot we do not have access to it's control inputs. However, this state formulation and the robot's kinematics are sufficient to extract the velocity estimate, as described by the differential equation which follows in (7).

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\psi}\\ \dot{v}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix}v\cos\psi\\ v\sin\psi\\ \omega\\ 0\\ 0\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\\ 0 & 0\\ 1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}\mathrm{w}_a\\ \mathrm{w}_\alpha\end{bmatrix} \tag{7}$$

In Equation (7), the linear and angular accelerations are modelled as the Gaussian random vector $[\mathrm{w}_a \quad \mathrm{w}_\alpha]^T$ with covariance $\mathbf{Q}$. In order to obtain the discrete movement model, a numerical method, such as Runge Kutta, can be used to approximate the differential equation.

Considering the VSS competition, as described in Section 1, the observation model is given by the vision system, with the camera positioned above the field. This system provides observations of the robot's position and orientation. However, this data is noisy, and we assumed this noise to be additive and Gaussian with covariance $\mathbf{R}$, as can be seen in Equation (8).

$$\mathbf{z}_k = \begin{bmatrix} x_k \\ y_k \\ \psi_k \end{bmatrix} + \begin{bmatrix} \mathrm{v}_{x,k} \\ \mathrm{v}_{y,k} \\ \mathrm{v}_{\psi,k} \end{bmatrix} \tag{8}$$

The second model for a differential drive robot, which will be called Double Integrator model, comes from a simpler formulation of its dynamics, as shown in Equation (9), already in the discrete form. In this case, the robot's state is $\mathbf{x}_k$=$[x_k \ y_k \ v_{x,k} \ v_{y,k}]^T$. With this model, also follows a simpler observation, given solely by the agent's position, as described in Equation (10).

In the VSS competition, it may not be possible to easily extract the opponent's orientation from the vision system, since the robot's color pattern used for identification can differ among different teams. This fact was the main motivation to elaborate a second stochastic model to deal with this possible limitation. Further we will also compare both models precision in the linear velocity estimate.

$$\begin{bmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ v_{x,k-1} \\ v_{y,k-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \begin{bmatrix} \mathrm{w}_{a,x} \\ \mathrm{w}_{a,y} \end{bmatrix} \tag{9}$$

$$\mathbf{z}_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix} + \begin{bmatrix} \mathrm{v}_{x,k} \\ \mathrm{v}_{y,k} \end{bmatrix} \tag{10}$$

4 Covariance Calibration

In Section 3, we saw that, for the proposed models, we have almost all of the parameters necessary for the Tracking. However, one challenge while implement tracking capabilities is how to adjust the covariance in order to have good results.

When the ground truth is available (i.e. some measurement that one can consider free from error), a procedure similar to the one described in (Burchardt et al., 2011) could be used. Defining a weighted norm $||\mathbf{x}||^2_\Sigma = \mathbf{x}^T\Sigma^{-1}\mathbf{x}$, in which Σ is a positive semi-definite matrix, we can define the total error of the tracker, given a constant set of observations and prior, as described in Equation (11). In this equation, N is the total number of time steps, $\mathbf{x}_{i,filter}$ is the estimated state by the tracking filter at time step i and $\mathbf{x}_{i,gt}$ is the ground truth state at the same time step.

$$E_{rr}(\mathbf{R},\mathbf{Q}) = \sum_{i=0}^{N} ||\mathbf{x}_{i,filter}(\mathbf{R},\mathbf{Q}) - \mathbf{x}_{i,gt}||^2_\Sigma \tag{11}$$

$$\mathbf{R}_{cal}, \mathbf{Q}_{cal} = \underset{\mathbf{R},\mathbf{Q}}{argmin}\ Err(\mathbf{R},\mathbf{Q}) \tag{12}$$

Then, to calibrate the covariances used by the filter processing real data, it would be sufficient to collect some observations and the ground truth data to choose $\mathbf{R}$ and $\mathbf{Q}$ according to Equation (12). To solve the minimization problem, we could, for instance use MATLAB's *fmincon* assuming a positive definite structure for both Q and R.

However, a very reliable tracking system to provide the ground truth may not be available. In order to use an idea similar to the one described above, we propose to find some signal that contains less error than the ones we obtained by the ground truth using signal processing theory.

Since signal processing theory is well established for 1 dimension signals, we decided to choose some one dimensional signal, which was, in this case, the linear velocity. We have redefined the error calculation as in Equation (13), in which v is the linear velocity of the robot ($v = \sqrt{v_x^2 + v_y^2}$).

$$E_{rr}(\mathbf{R},\mathbf{Q}) = \sum_{i=0}^{N} (v_{i,filter}(\mathbf{R},\mathbf{Q}) - v_{i,gt})^2 \tag{13}$$

Because of the lesser noisy in the velocity information, a low pass filter was used on the raw velocity information extracted by the computer vision system. Notice that the robot is band-limited and the velocity is obtained by numerical derivation of the already noisy raw data, which amplifies high-frequency noise. Then, since we could have some sample velocity signals before hand, it was possible to process it offline. By the use of a non-causal high-order non pass filter with zero group delay, a much smoother signal was obtained without adding any delay to the shape of the signal (Oppenheim and Schafer, 1989).

5 Simulation and Results

In order to verify the performance of the tracking techniques, we applied the Kalman filters using vision information collected from a test with one differential drive robot from ITAndroids' VSS team. Also, a simulation was realized to compare tracking and vision performance.

5.1 Results with real robots

In the test using the robot, we commanded movements along the VSS's field and extracted the

Cartesian position and orientation provided by the vision algorithm. Using this data, we implemented the EKF and UKF for the Unicycle model described in Section 3, with state $\mathbf{x}_k$=$[x_k\ y_k\ \psi_k\ v_k\ \omega_k]^T$, and the linear Kalman Filter for the Double Integrator model, with state $\mathbf{x}_k$=$[x_k\ y_k\ v_{x,k}\ v_{y,k}\ \psi_k\ \omega_k]^T$. The trajectory followed by the robot can be seen in figure 2, which contains the observed and estimated position by EKF and UKF for the Unicycle model.

Figure 3 shows the linear (figure 3(a)) and angular (figure 3(b)) speeds obtained by the EKF and UKF for the Unicycle model, as well as the speeds computed by numerical differentiation of the position and orientation from the measurement. In the same way, figure 4 presents the velocity in x-axis calculated from the measurement and by the Kalman filter for the Double Integrator model. In that case, the velocity obtained in y-axis has a similar behaviour. The covariances are the ones obtained by the procedure described in Section 4. The experiments results can also be watched in https://youtu.be/3WVtdEw4jlE.

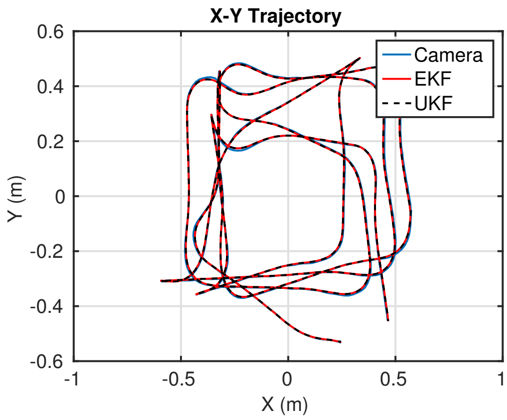

Figure 2: Observed and estimated trajectories using EKF and UKF

5.2 Simulation

Despite the good results observed in the experiment with the robot and its real data, we can not extract an exact evaluation of the precision improvement obtained by the tracker technique using this procedure alone. In the carried out tests, we do not have a Ground Truth information of the robot's state, and consequently it is not possible to directly compare the tracking information to those from the camera.

Hence, we also made simulations to test the tracking algorithms. As our purpose is also compare both described models in the velocity estimate, we used all filters for the same system simulation in order to compare the estimates with the same ground truth. In that process, the evolution of the actual state is given by the Unicycle model, since this model is closer to the real dynamics of

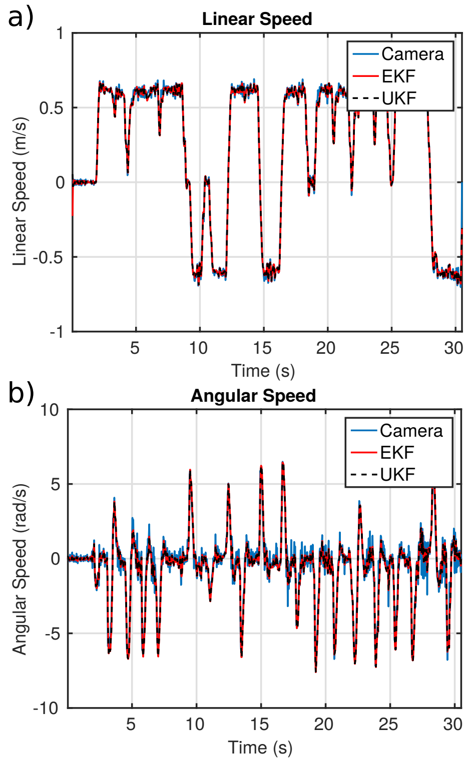

Figure 3: Computed velocities from observation, EKF and UKF:(a)linear velocity, (b)angular velocity

the robot. In order to apply the Kalman filter using the Double Integrator model, and EKF and UKF using the Unicycle model for the same actual and observed data, we used the covariance of w_a, already obtained by optimization, in w_α, $\mathrm{w}_{v,x}$ and $\mathrm{w}_{v,y}$ for the filters update and system evolution. The covariance matrix $\mathbf{R}$ is the same used in the real data test.

The simulations lasted total time of $30s$, each time step is constant $T = \frac{1}{30}s$, which correspond to the average camera frequency (30 FPS). In order to estimate the RMSE, 1000 Monte Carlo realizations were performed.

Since position information from the camera is already very precise, again we will focus on the velocities estimate. The simulation results can be found in Figure 5, which shows in subfigure 5(a) the RMSE for linear velocity, calculated from differentiation of observation, EKF, UKF and KF for their correspondent models. In subfigure 5(b) one can see the RMSE for angular velocity, computed only by observation and estimates from EKF and UKF.

5.3 Discussions

All models and filters were very successful in obtaining a good estimate for linear velocity. In the Unicycle dynamic model, we have a more complete state including either orientation and angular velocity. In addition, this dynamic descrip-

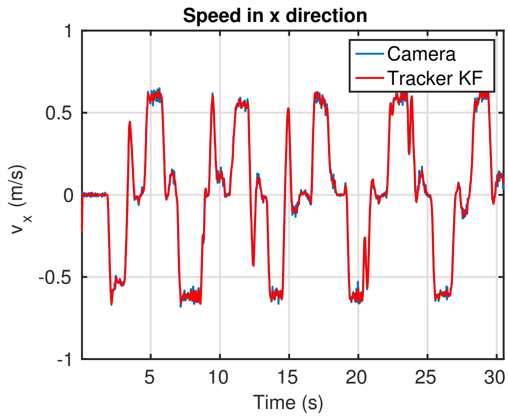

Figure 4: velocity in x-axis given by observation and Kalman Filter

tion is more faithful to the real one performed by the differential robot. For the tracking test, EKF and UKF presented the same results, indicating that there is no necessity for a more complex technique such as UKF, since we have a model without strong non-linearities and that no analytical complications are present. Therefore, EKF would be preferable due to the simplicity and computational cost.

In the double integrator model using linear Kalman filter, despite the lower accuracy to represent the differential robot dynamics, we might claim that it presented almost the same estimation performance in comparison to EKF and UKF in the Unicycle model. In the simulation performed, EKF demonstrated an average RMSE only 1.5% smaller than KF. Moreover, from this model results, we can conclude that it would successfully estimate velocity for the case in which we do not have the robot's orientation.

6 Conclusions

From the analysis in Section 5, it is possible to state that the stochastic model proposed, along with Kalman Filtering is able to provide satisfactory results in tracking a differential drive robot and, especially, in estimating its velocity. In addition, we have introduced a method to calibrate the filter's covariance in the absence of ground truth information.

The tracking results obtained with actual robots confirm that those techniques can be successfully applied in practical and real world problems, while the Monte Carlo simulation provides a better assessment of the filter's performance compared to the observations solely. We have also shown that in case of using a worse sensor than the computer vision of the VSS league, the tracking can provide even larger relative error reduction when compared to the raw sensor information.

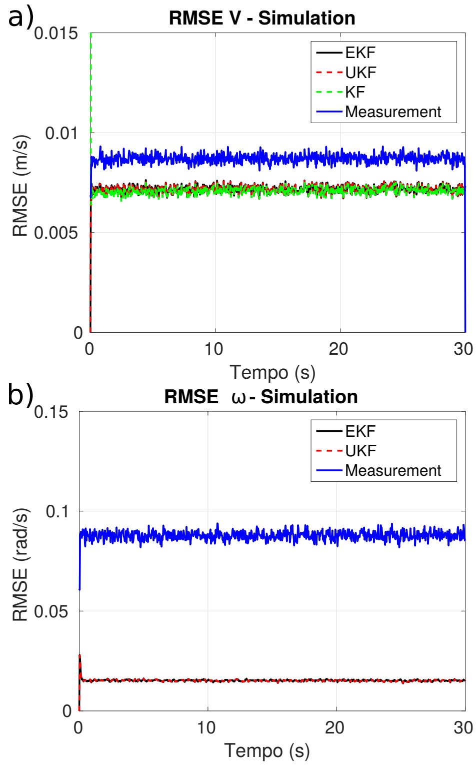

Figure 5: RMSE for the velocities calculated from observations, EKF and UKF: (a)linear velocity, (b)angular velocity

References

Arulampalam, S., Maskell, S., Gordon, N. and Clapp, T. (2001). A tutorial on particle filters for on-line non-linear/non-gaussian bayesian tracking, Vol. 20.

Brookner, E. (1998). *Tracking and Kalman Filtering Made Easy*, John Wiley and Sons, Inc.

Bruno, M. G. S. and Pavlov, A. (2004). Improved particle filters for ballistic target tracking.

Burchardt, A., Laue, T. and Röfer, T. (2011). Optimizing particle filter parameters for self-localization, *RoboCup 2010: Robot Soccer World Cup XIV*, Springer, pp. 145–156.

Julier, S. J. and Uhlmann, J. K. (2004). Unscented Filtering and Nonlinear Estimation, *Proceedings of the IEEE*, Vol. 92.

Oppenheim, A. V. and Schafer, R. W. (1989). *Discrete-time Signal Processing*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Palumbo, N. F., Harrison, G. A., Blauwkamp, R. A. and Marquart, J. K. (2010). Guidance Filter Fundamentals, *Johns Hopkins APL TEchnical Digest*, Vol. 29.

S. Thrun, W. B. and Fox, D. (2005). *Probabilistic Robotics*, MIT Press.