Comparison and Evaluation of Advanced Motion Models for Vehicle Tracking

Robin Schubert, Eric Richter, Gerd Wanielik
Professorship of Communications Engineering
Chemnitz University of Technology
Reichenhainer Straße 70, 09126 Chemnitz, Germany
Email: {robin.schubert, eric.richter, gerd.wanielik}@etit.tu-chemnitz.de

***Abstract*— The estimation of a vehicle's dynamic state is one of the most fundamental data fusion tasks for intelligent traffic applications. For that, motion models are applied in order to increase the accuracy and robustness of the estimation. This paper surveys numerous (especially curvilinear) models and compares their performance using a tracking tasks which includes the fusion of GPS and odometry data with an Unscented Kalman Filter. For evaluation purposes, a highly accurate reference trajectory has been recorded using an RTK-supported DGPS receiver. With this ground truth data, the performance of the models is evaluated in different scenarios and driving situations.**

Keywords: Vehicle Tracking, Motion Models, UKF

I. Introduction

Vehicle tracking is one of the most important data fusion tasks for Intelligent Transportation Systems (ITS). Especially for advanced driver assistance systems such as Collision Avoidance/Collision Mitigation (CA/CM), Adaptive Cruise Control (ACC), Stop-and-Go-Assistant, or Blind Spot Detection, a reliable estimation of other vehicles' positions is one of the most critical requirements.

In order to increase the stability and accuracy of the estimation, the vehicles are mostly assumed to comply with certain *motion models* which describe their dynamic behavior. Another advantage of this approach is the ability to predict the vehicle's position in the future (which can for instance be used to calculate a collision probability). From the data fusion point of view, the task is to estimate the parameters of the model – taking into account all available observations. The most common approach for this task is the Kalman Filter or one of its derivates [1].

The application of motion models has been intensively studied for a variety of ITS applications, for instance radar tracking [2] or navigation [3]. However, even applications which are from a superficial point of view not concerned by vehicle tracking often require a reliable estimation of the ego vehicle's motion in order to compensate estimates of tracked objects accordingly (an example which illustrates this is motion based pedestrian recognition [4]). Thus, the term *vehicle tracking* in this paper refers to the task of estimating the model parameters of either the ego vehicle or vehicles in its surrounding.

In the past, numerous motion models (with different degrees of complexity) have been proposed for this task. Some authors also compared different motion models for a certain applications in a rather general way using simulated data (e. g. [5]). However, the question which motion model is most suitable for describing vehicles' motions in certain scenarios has not yet been sufficiently answered and will therefore be the subject of this paper. In particular, an evaluation approach is proposed which is based on the combination of GPS and odometry measurements. By comparing the estimates of every model with a highly accurate reference trajectory, the filters' performances can be compared and evaluated.

The paper is organized as follows: Section II surveys the most common motion models and their state transition equations. In the following section, the methodology for evaluating the models is described. Finally, the results of the comparison are presented and discussed in section IV.

II. Survey on Motion Models

A. Systematization

As indicated above, the models proposed in literature are numerous. A first systematization can be achieved by defining different levels of complexity. At the lower end of such a scale, *linear motion models* are situated. These models assume a *constant velocity* (CV) or a *constant acceleration* (CA). Their major advantage is the *linearity* of the state transition equation which allows an optimal propagation of the state probability distribution.[1] On the other hand, these models assume straight motions and are thus not able to take rotations (especially the yaw rate) into account.

A second level of complexity can be defined by taking rotations around the z-axis into account. The resulting models are sometimes referred to as *curvilinear* models. They can be further divided by the state variables which are assumed to be constant. The most simple model of this level is the *Constant Turn Rate and Velocity* (CTRV) model, which is commonly used for airborne tracking systems [6].[2] By defining the derivative of the velocity as the constant variable, the *Constant Turn Rate and Acceleration* (CTRA) model can be derived. Both CTRV and CTRA assume that there is no

[1] However, note that the measurement equation is necessarily nonlinear if the orientation angle is included in the state vector.

[2] Note that in literature, this model is sometimes referred to as *CTR*. However, in order to obtain a consistent nomenclature, CTRV will be consequently used throughout this paper.

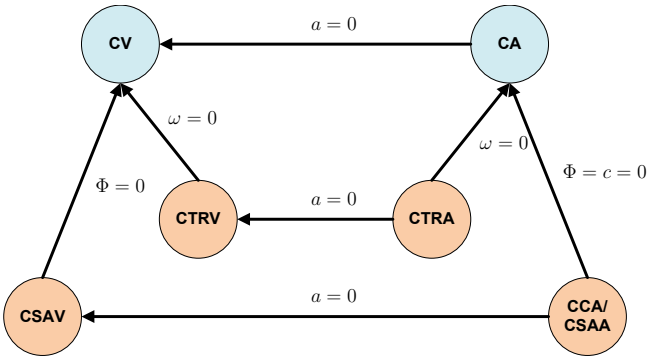

Fig. 1. Overview about linear and curvilinear motion models. Every sophisticated model can be transformed into a simpler one by setting one state veriable to zero.

correlation between the velocity v and the yaw rate ω. As a consequence, disturbed yaw rate measurements can change the yaw angle of the vehicle even if it is not moving.

In order to avoid this problem, the correlation between v and ω can be modeled by using the steering angle Φ[3] as constant variable and derive the yaw rate from v and Φ. The resulting model is called *Constant Steering Angle and Velocity* (CSAV). Again, the velocity can be assumed to change linearly, which leads to the *Constant Curvature and Acceleration* (CCA) model.[4] The connections between all models described so far are illustrated in figure 1.

From a geometrical point of view, nearly all curvilinear models are assuming that the vehicle is moving on a circular trajectory (either with a constant velocity or acceleration). The only exception is the CTRA model which models a linear variation of the curvature and thus assumes that the vehicle is following a clothoid.

While in theory curvilinear models describe the motion of road vehicles very accurately, errors may result from highly dynamic effects such as drifting or skidding. While models which are able to cope with such effects do exist (e. g. [7]), they will not be considered here for two reasons: Firstly, most ITS applications are designed for scenarios with non-critical dynamics. Furthermore, the information which are necessary for estimating the additional parameters (e. g. slip from every tire, lateral acceleration) are not observable by exteroceptive sensors. Thus, such models can be used for estimating the ego vehicle's motion, only.

B. State Transition Equations

Many of the described models (with the exception of CCA) are well-known and will thus be treated very briefly. Further details can be found in [1].

[3]This angle is defined between the axis of motion and the direction of the front wheels.

[4]If the steering angle would be used as a state variable instead of the curvature, the model could also be named *Constant Steering Angle and Velocity* (CSSA). From an algorithmic point of view, however, both names refer to the same model.

1) CV: As the CV model with the state space

$$\vec{x}(t) = \begin{pmatrix} x & v_x & y & v_y \end{pmatrix}^T \tag{1}$$

is a linear motion model, the linear state transition

$$\vec{x}(t+T) = A(t+T)\vec{x}(t) \tag{2}$$

is substituted by the state transition function vector

$$\vec{x}(t+T) = \begin{pmatrix} x(t) + Tv_x \\ v_x \\ y(t) + Tv_y \\ v_y \end{pmatrix} \tag{3}$$

in order to use it within the Unscented Kalman Filter framework.

2) CTRV: The state space

$$\vec{x}(t) = \begin{pmatrix} x & y & \theta & v & w \end{pmatrix}^T \tag{4}$$

can be transformed by the non-linear state transition

$$\vec{x}(t+T) = \begin{pmatrix} \frac{v}{\omega}\sin(\omega T + \theta) - \frac{v}{\omega}\sin(\theta) + x(t) \\ -\frac{v}{\omega}\cos(\omega T + \theta) + \frac{v}{\omega}\sin(\theta) + y(t) \\ \omega T + \theta \\ v \\ \omega \end{pmatrix}. \tag{5}$$

3) CTRA: The state space of this models expands the last one by a:

$$\vec{x}(t) = \begin{pmatrix} x & y & \theta & v & a & w \end{pmatrix}^T. \tag{6}$$

The state transition equation for this model is:

$$\vec{x}(t+T) = \begin{pmatrix} x(t+T) \\ y(t+T) \\ \theta(t+T) \\ v(t+T) \\ a \\ \omega \end{pmatrix} = \vec{x}(t) + \begin{pmatrix} \Delta x(T) \\ \Delta y(T) \\ \omega T \\ aT \\ 0 \\ 0 \end{pmatrix}, \tag{7}$$

with

$$\begin{aligned} \Delta x(T) = \; & \tfrac{1}{\omega^2}[(v(t)\omega + a\omega T)\sin(\theta(t) + \omega T) \\ & + a\cos(\theta(t) + \omega T) \\ & - v(t)\omega\sin\theta(t) - a\cos\theta(t)] \end{aligned} \tag{8}$$

and

$$\begin{aligned} \Delta y(T) = \; & \tfrac{1}{\omega^2}[(-v(t)\omega - a\omega T)\cos(\theta(t) + \omega T) \\ & + a\sin(\theta(t) + \omega T) \\ & + v(t)\omega\cos\theta(t) - a\sin\theta(t)] \end{aligned}. \tag{9}$$

4) CCA: The state space

$$\vec{x}(t) = \begin{pmatrix} x & y & \theta & v & a & c \end{pmatrix}^T \tag{10}$$

is similar the one of the CTRA model, except that the yaw rate ω is replaced by the curvature $c = R^{-1}$, where R represents the radius the vehicle is currently driving. Because of

$$R = \frac{1}{c} = -\frac{v(t)}{\omega(t)} = const., \tag{11}$$

and

$$v(t) = v(t_0) - at, \tag{12}$$

the yaw rate becomes a function of time

$$\omega(t) = (-v(t_0) - at)c. \tag{13}$$

The continuous system can be described by

$$\vec{\dot{x}}(t) = \begin{pmatrix} v(t)\cos\left(\omega(t)t + \theta(t_0)\right) \\ v(t)\sin\left(\omega(t)t + \theta(t_0)\right) \\ \omega(t)t \\ a \\ 0 \\ 0 \end{pmatrix}. \tag{14}$$

Using the equations 12 and 13, the final system follows to

$$\vec{\dot{x}}(t) = \begin{pmatrix} (v_0 + at)\cos((-v_0 - at)ct + \theta_0) \\ (v_0 + at)\sin((-v_0 - at)ct + \theta_0) \\ (-v_0 - at)c \\ a \\ 0 \\ 0 \end{pmatrix} \tag{15}$$

The discrete state transition equation arises from integrating the continuous one

$$\vec{x}(t+T) = \int_t^{t+T} \vec{\dot{x}}(t)\,dt + \vec{x}(t), \tag{16}$$

which leads to the state transition equation 17 with

$$\gamma_1 = \frac{1}{4a}\left(cv^2 + 4a\theta\right), \tag{18}$$

$$\gamma_2 = cTv + cT^2a - \theta, \tag{19}$$

$$\eta = \sqrt{2\pi}vc, \tag{20}$$

$$\zeta_1 = (2aT + v)\sqrt{\frac{c}{2a\pi}}, \tag{21}$$

$$\zeta_2 = v\sqrt{\frac{c}{2a\pi}}, \tag{22}$$

$$C(\zeta) = \int_0^{\zeta} \cos\left(\frac{\pi}{2}x^2\right)dx, \tag{23}$$

and

$$S(\zeta) = \int_0^{\zeta} \sin\left(\frac{\pi}{2}x^2\right)dx. \tag{24}$$

Since equations 23 and 24 represent the fresnel integrals [8], a numerical approximation is used for calculating their values.

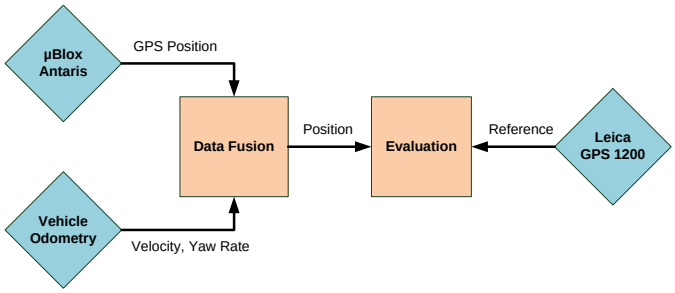

Fig. 2. The experimental setup for the model evaluation. The odometry was obtained by internal vehicle sensors.

III. Methodology

A. Experimental Setup

In order to perform a motion model evaluation on the basis of real experimental data, the system setup shown in figure 2 was used. The data fusion module estimates the current position of the ego vehicle; taking into account GPS/EGNOS, velocity, and yaw rate measurements (the latter are obtained via CAN bus from the internal vehicle sensors).

In addition, a DGPS receiver with RTK capabilities is used to obtain a highly accurate reference trajectory. Under optimal conditions, the accuracy of the position measurements is in the range of a few centimeters. An integrated monitoring algorithm reports if this accuracy cannot be achieved; such measurements are not used for the evaluation.

Several test drives have been performed using the test vehicle *Carai* (compare figure 3). Different routes have been chosen to reflect typical scenarios for ITS applications, in particular urban environments and highways (see figure 4).

B. Model Choice

For the sake of clarity, not all models introduced in section II are included in the evaluation. It is rather the aim of this paper to bilaterally compare certain models in order to answer particular questions. In fact, the following issues shall be analyzed:

- The influence of explicitly modeling the yaw rate – this can be investigated by comparing CV and CTRV.
- The influence of modeling the acceleration (by comparing CTRV and CTRA).
- The influence of modeling the correlation between v and ω (by comparing CTRA and CCA).

$$\vec{x}(t+T) = \begin{bmatrix} x + \frac{\eta\cos(\gamma_1)C(\zeta_1) + \eta\sin(\gamma_1)S(\zeta_1) - \eta\cos(\gamma_1)C(\zeta_2) - \eta\sin(\gamma_1)S(\zeta_2) + 2\sin(\gamma_2)\sqrt{ac} + 2\sin(\theta)\sqrt{ac}}{4\sqrt{ac}c} \\ y + \frac{-\eta\cos(\gamma_1)S(\zeta_1) + \eta\sin(\gamma_1)C(\zeta_1) - \eta\sin(\gamma_1)C(\zeta_2) + \eta\cos(\gamma_1)S(\zeta_2) + 2\cos(\gamma_2)\sqrt{ac} - 2\cos(\theta)\sqrt{ac}}{4\sqrt{ac}c} \\ -\frac{1}{2}cT^2a - cTv + \theta \\ aT + v \\ a \\ c \end{bmatrix} \tag{17}$$

Fig. 3. The test vehicle *Carai* which was used for measurement recording.

Fig. 4. Bird eye's view on a sample scenario. The reference trajectory is drawn blue, whereas CTRV and CTRA are yellow and green, respectively.

Thus, only four models are analyzed: CV, CTRV, CTRA, and CCA.

C. Filter Algorithm

1) The Unscented Kalman Filter: As the aim of this paper is to compare different *models*, the same filter is used for the data fusion. As at least the measurement equations of all models are nonlinear, the *Unscented Kalman Filter* (UKF) was chosen for that task. It has been shown that this filter is able to perform nonlinear propagations of probability distributions more accurately than the linearized Extended Kalman Filter (EKF) [9]. The fundamentals of the UKF will be very briefly summarized in the following - more detailed descriptions can be found (among others) in [10].

The UKF is an unbiased, minimum-mean squared error estimator of a dynamic system with the state vector $\vec{x}$ and the covariance matrix P. In general, the structure of the UKF is similar to the well-known Extended Kalman Filter (EKF), that is, it predicts the state of the system using a (typically nonlinear) state transition equation $f(\vec{x})$ and corrects this prediction by incorporating observations which may come from different sensors.

The particularity of the UKF derives from the fact that for predicting the state vector of the system, not only one single value, but a whole probability distribution has to be transformed. While this is an easy task in the linear case, it causes significant problems if the state transition equation is nonlinear. The EKF's approach to that issue is a linearization of $f(\vec{x})$ around the current value of $\vec{x}$. The covariance matrix can then be predicted using the Jacobian of $f(\vec{x})$.

However, this approach has the unpleasing property of underestimating the uncertainty in many cases – in particular when transforming polar into cartesian coordinates and *vice versa* [11]. Although that property can partly be compensated by adding more process noise, it generally deteriorates the performance of the filter and in the worst case causes it to divert from the true value.

Another approach for transforming probability distributions in a nonlinear manner is to sample the distribution and transform the samples separately. If the samples are drawn randomly, this approach is called particle filter [12]. While this filter yields good tracking results, it requires a large amount of computational resources due to the large number of particles which are necessary to adequately represent the distribution. Other approaches are thus using deterministically chosen samples to transform the probability distribution. One special case of those so called Sigma Point Kalman Filters is the Unscented Kalman Filter.

The UKF performs the so called Unscented Transformation in order to calculate the sigma points (i. e. the samples) from the probability distribution. In particular, the sigma points at the discrete time stage $k-1$ are obtained by

$$\chi_{k-1} = \left[\vec{x}_{k-1} \quad \vec{x}_{k-1} + \gamma\sqrt{P_{k-1}} \quad \vec{x}_{k-1} - \gamma\sqrt{P_{k-1}}\right], \quad (25)$$

where L is the size of the state vector and $\gamma = \sqrt{\alpha^2(L+\kappa)}$ is a scaling parameter.[5]. $\sqrt{P}$ denotes the *Cholesky decomposition* of the covariance matrix, i. e. every column of the lower-triangular Cholesky-decomposed P-matrix is added to and subtracted from the mean. After applying the state transition equation to the sigma points, that is $\chi_k = f(\chi_{k-1})$, the predicted mean and covariance of the state vector can be calculated as follows:

$$\vec{x}_k^* = \sum_{i=0}^{2L} W_i^m \chi_{i,k} \quad (26)$$

$$P_k^* = \sum_{i=0}^{2L} W_i^c (\chi_{i,k} - \vec{x}_k^*)(\chi_{i,k} - \vec{x}_k^*)^T \quad (27)$$

[5]There are three scaling parameters which determine the behaviour of the filter: α determines the spread of the sigma points around the mean value and is usually set to a small value (for the application described here, it is set to 10^{-5}). β can be set to 2 if Gaussian distributions are assumed, while $\kappa = 3 - L$ (see [10] for details).

In these equations, the weights W_i can be derived from the above-mentioned scaling parameters:

$$\begin{aligned} W_0^m &= \frac{\gamma^2 - L}{\gamma^2}, \\ W_0^c &= \frac{\gamma^2 - L}{\gamma^2} + 1 - \alpha^2 + \beta, \\ W_i^m = W_i^c &= \frac{1}{2\gamma^2}, \qquad i \neq 0 \end{aligned} \tag{28}$$

While the update step is now complete, the state still has to be corrected by the observations. For that, it is at first transformed into the observation space using the (in general also nonlinear) function h in order to obtain the expected observations $\vec{y}^*$ and their covariance U^*:

$$\Upsilon_{k} = h(\chi_{k}) \tag{29}$$

$$\vec{y}_k^* = \sum_{i=0}^{2L} W_i^m \Upsilon_{i,k} \tag{30}$$

$$U_k^* = \sum_{i=0}^{2L} W_i^c (\Upsilon_{i,k} - \vec{y}_k^*)(\Upsilon_{i,k} - \vec{y}_k^*)^T \tag{31}$$

Finally, the predicted and the observed measurements can be combined in the fusion step of the UKF:

$$P_{x_k,y_k} = \sum_{i=0}^{2L} W_i^c (\chi_{i,k} - \vec{x}_k^*)(\Upsilon_{i,k} - \vec{y}_k^*)^T \tag{32}$$

$$K_{k} = P_{x_k,y_k} U_k^{*-1}, \tag{33}$$

$$x_{k} = x_k^* + K_{k}(\vec{y}_{k} - \vec{y}_k^*), \tag{34}$$

$$P_{k} = P_k^* + K_{k} U_k^* K_k^T \tag{35}$$

2) Noise Considerations: It should be noted that neither the process nor the measurement noise have been mentioned so far. They can be incorporated into the state estimation in two different ways: For purely additive noise with zero mean, the process noise covariance matrix Q and the measurement noise covariance matrix R can simply be added to equation 27 and 31, respectively. In this paper, such an approach is used for the measurement noise only. For the process noise, however, this approach does not appear reasonable, as this would mean to derive a time-discrete covariance matrix by *linearization* – whose avoidance is one of the main advantages of the UKF. Instead, the state vector is augmented by some noise variables e_n as proposed in [10]. Thus, the noise is included into the estimation process. Apart from avoiding linearization, this approach is also able to cope with non-additive or non-white noise.

While the measurement noise can be chosen identical for every model, difficulties arise from the process noise parameters. This is due to the fact that this noise refers to different state variables in every model. For example, the process noise in the CV model disturbs v_x and v_y, while for the CCA model c and a are affected. In order to cope with that problem, the following procedure is proposed: If different models contain the same error variables (e. g. e_a in all models which assume constant acceleration), the values are chosen identically. Unique noise parameters of single models are chosen in a way which maximizes the filter performance. The intuition behind this proposal is that the performance of the filters should not artificially be decreased. The concrete values of the noise terms can be found in table I.

TABLE I
NOISE PARAMETERS FOR THE UNSCENTED KALMAN FILTER

Error variable	Standard Derivation
e_{v_x}	0.5
e_{v_y}	0.5
e_v	0.5
e_w	0.25
e_a	0.5
e_c	0.25

TABLE II
ERROR COMPARISON FOR URBAN AND HIGHWAY SCENARIO

			Model			
		RMS [m]	CV	CTRV	CTRA	CCA
Scenario	Urban	Euclidian	3.17	2.36	1.85	2.49
		Lateral	2.32	1.76	1.31	1.68
		Longitudinal	2.16	1.58	1.31	1.84
	Highway	Euclidian	3.89	3.98	3.35	3.36
		Lateral	1.67	1.52	1.35	1.35
		Longitudinal	3.52	3.68	3.07	3.08

IV. RESULTS

Table II shows the results of comparing all presented models for the two scenarios urban and highway. It is distinguished between the euclidean, the lateral, and longitudinal error, respectively. It can be seen that the sophisticated models CTRV, CTRA and CCA provide a better performance than the simple CV model in almost every case. Furthermore, the incorporation of the acceleration additionally improves the overall tracking performance.

In order to evaluate the influence of modeling the yaw rate and using it as an additional observation, the CV and CTRV models have been compared using scenes with a high curvature only ($\omega > 2\,^\circ/s$) (compare figure 5). The diagram shows that the CV model generates large position errors due to high curvature, whereas the CTRV model is able to provide a better estimation.

The advantage of using acceleration enhanced models like CTRA is illustrated in figure 6. There, only situations with accelerations higher than $0.5\,m/s^2$ are included. Especially in such high acceleration situations, the CTRA model performs much better than the CTRV model.

As shown in figure 7, it appears that there is no significant difference between the CCA and CTRA model. Due to the much higher calculation effort the CTRA model should be used instead of the CCA model.

V. CONCLUSIONS

In this paper, it has been shown that the choice of an appropriate model can significantly increase the performance

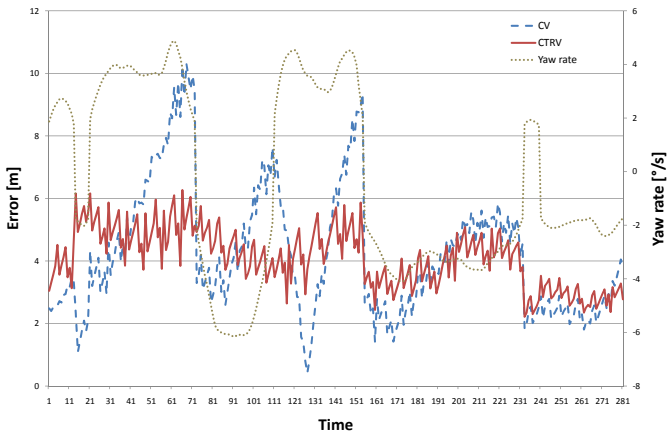

Fig. 5. Euclidean error of CV and CTRV model in a highway scenario

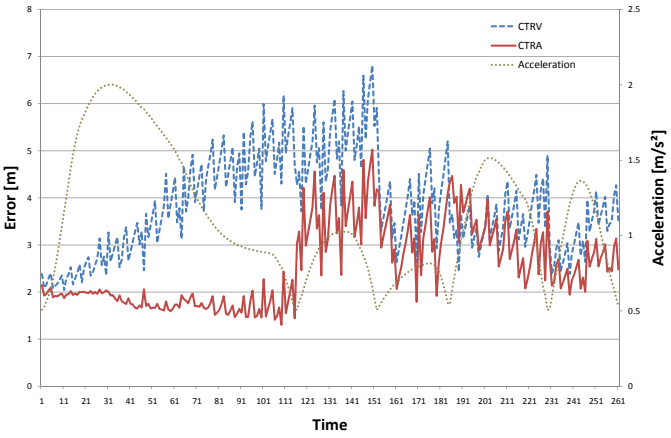

Fig. 6. Euclidean error of CTRV and CTRA model in a highway scenario

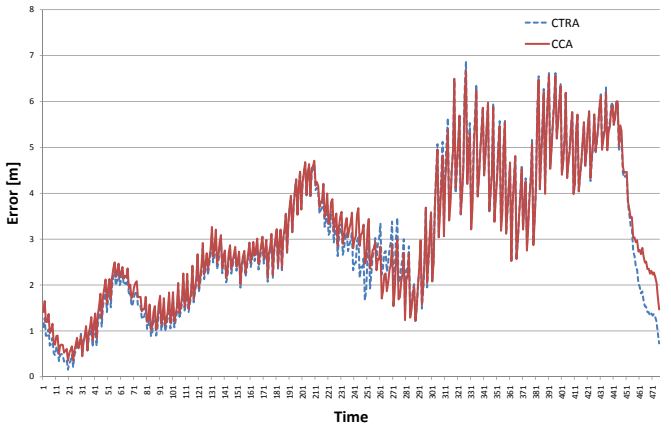

Fig. 7. Euclidean error of CTRA and CCA model in an urban scenario

of a vehicular tracking system. in general, more sophisticated models outperform simpler ones, especially in situations where the assumptions of the simple models are no longer true. However, it appears that making models more sophisticated does not lead to better performances in any case, as the example of the CCA model shows.

In order to further improve tracking systems, it is often beneficial to combine the strengths of different models and simultaneously avoid their weaknesses. An algorithm which follows this aim has been proposed by [1] and is called the *Interacting Multiple Model* filter. However, the increased performance of such a filter is always bought with computational costs.

REFERENCES

[1] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor tracking: Principles and Techniques*. Storrs, CT: YBS Publishing, 1995.

[2] E. Richter, R. Schubert, and G. Wanielik, "Advanced Filtering Techniques for Multisensor Vehicle Tracking," in *Procs. of IEEE Intelligent Vehicle Symposium (under review)*, 2008.

[3] R. Schubert, N. Mattern, and G. Wanielik, "An evaluation of nonlinear filtering algorithms for integrating gnss and inertial measurements," in *Proc. of ION/IEEE Positioning, Localization and Navigation Symposium PLANS (accepted for publication)*, 2008.

[4] B. Fardi, I. Seifert, G. Wanielik, and J. Gayko, "Motion-based pedestrian recognition from a moving vehicle," in *Proc. IEEE Intelligent Vehicles Symposium*, 2006, pp. 219–224.

[5] M. Tsogas, A. Polychronopoulos, and A. Amditis, "Unscented Kalman Filter Design for Curvilinear Motion Models Suitable for Automotive Safety Applications," in *Proc. 8th International Conference on Information Fusion*. IEEE, 2005, pp. 1295–1302.

[6] S. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Artech House, 1999.

[7] R. Pepy, A. Lambert, and H. Mounier, "Reducing navigation errors by planning with realistic vehicle model," in *Intelligent Vehicles Symposium, 2006 IEEE*, 13-15 June 2006, pp. 300–307.

[8] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, ninth dover printing, tenth gpo printing ed. New York: Dover, 1964.

[9] S. Julier and J. K. Uhlmann, "A general method for approximating nonlinear transformations of probability distributions," RRG, Dept. of Engineering Science, University of Oxford," technical report, November 1996. [Online]. Available: http://www.cs.unc.edu/~welch/kalman/media/pdf/Julier1997_SPIE_KF.pdf

[10] E. A. Wan and R. van der Merwe, *Kalman Filtering and Neural Networks*, ser. Adaptive and Learning Systems for Signal Processing, Communications, and Control. John Wiley & Sons, Inc., 2001, ch. The Unscented Kalman Filter, pp. 221–280.

[11] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004. [Online]. Available: http://www.cs.ubc.ca/~murphyk/Papers/Julier_Uhlmann_mar04.pdf

[12] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter – Particle Filters for Tracking Applications*. Artech House, 2004.