A Brief Tutorial On Recursive Estimation With Examples From Intelligent Vehicle Applications (Part I): Basic Spirit And Utilities

Hao Li

Abstract

Estimation is an indispensable process for an ocean of applications, which are rooted in various domains including engineering, economy, medicine, etc. Recursive estimation is an important type of estimation, especially for online or real-time applications. In this brief tutorial, we will explain the utilities, the basic spirit, and some common methods of recursive estimation, with concrete examples from intelligent vehicle applications.

Keywords: recursive estimation, Bayesian inference, Kalman filter (KF), intelligent vehicles

1 Introduction

Estimation, simply speaking, is a process of "revealing" ("finding" etc) the true value of certain entity (or entities) that we care about in certain activity (or activities). Even more generally and abstractly speaking, estimation can be regarded as a process of trying to know our world. In this sense, estimation is a ubiquitous process for human activities. No matter which activity we are going to carry out, we will normally not act blindly but act according to our evaluation of things related to our activity. Take the ordinary and routine behavior, walking, as an example: during a walking process, we will figure out unceasingly a navigable (and often as short as possible) path according

to our observation of environment objects . In other words, we base our walking behavior unceasingly on estimation of the environment in which we are walking.

In above explanations, the intuitive words "entities that we care about" are more formally referred to as the term **state** by researchers. A state can refer to the pose (position and orientation) of an airplane; it can refer to the poses of a group of airplanes; it can refer to the temperature of a patient; it can refer to the trading volume of a capital market, etc. In intelligent vehicle applications (to which the examples in this brief tutorial are related), a state can refer to the vehicle pose [1] [2] [3] [4], can refer to certain info of specific objects (such as lane marks and surrounding vehicles) [5] [6] [7], can refer to a generic representation of the local environment of a vehicle [8] [9] etc. What a state actually refers to depends on concrete applications.

Like **state**, another important concept well related to estimation is **system model** (we examine entities from system perspective). A system model of a state, is a (mathematical) model describing the constraints that the state should satisfy as it evolves, or in other words, a (mathematical) model describing the laws (physical, chemical, biological etc) that govern the evolution or change of the state. Strictly speaking, we do not necessarily need a system model to estimate a state, yet a better system model can help yield a better estimate of the state. Consider the following problem as an example: if we know certain object was in Paris one hour ago, and where may be the object now? Given no specific information on the object, we may only establish for its movement a weak system model no better than a random guess model—Imagine that the object is a kind of hyper-developed alien who masters transportation devices that move even much faster than light, and it may appear potentially anywhere in the whole universe—If we know the object is an airplane, we may establish a more specific system model considering the speed limitation of a normal airplane; then our answer is that it may be somewhere in Europe and the answer becomes much better. If we further know this airplane has been heading east since one hour ago, we may establish a even detailed system model considering its moving direction; then our answer is that it may be somewhere in Germany and the answer becomes even better. Generally for a state, the better system model we have, the better estimate of the state we may also obtain.

The third concept well related to estimation is **measurement** a.k.a. **observation**. A measurement on a state, as the term itself implies literally, is a measured value of the state or a measured value related to the state.

Measurements on a state can be provided by specially designed instruments. For example, an odometer, a speedometer, and an accelerometer installed on a vehicle output respectively measurements on the traveled distance, the speed, and the acceleration of the vehicle.

One may naturally ask: given that we can have measurements on a state, do we still need to "estimate" the state and why? The answer is that we do need to estimate the state. Reasons are two-folds First, measurements usually have measurement errors or noises and may not reflect the true value of a state. So we need an estimation technique to reveal the true value of the state—although this will never be strictly achieved, yet we need to try to obtain an estimate as precise as possible. Second, measurements sometimes only reflect a state partially. In this case, we need an estimation technique to reveal the entire state including the state part that is not directly measurable (or directly observable) out of partial measurements on the state. An typical example is a process of vehicle localization, where we need to estimate the position as well as the orientation of a vehicle out of measurements only on the position of the vehicle. Details of the example will be given in later sections.

As we have the concept measurement now, we can further introduce the forth important concept **measurement model**. A measurement model describes the relationship between the measurement and the state. Simply speaking, a measurement model "predicts" what the measurement on a state might be given that the state is known. This logic seems not to be consistent with our intuition, because if we do have the true value of the state, it would be meaningless to examine the so-called measurement model and other "trouble" stuff related to estimation. To explain the utility of the measurement model, we give a daily-life experience as a simple analogous example: suppose a friend would come to your home for dinner in the evening if it did not rain, and would not if it rained. In the evening, this friend has not come. Question: what is the weather? One can easily give the answer that it rains. In this example, the weather can be regarded as the state, the friend's choice can be regarded as the measurement on this "state". The relationship between the weather and the friend's choice is what we mean a "measurement model" here. As we can see, the "measurement model" helps our inference on the "state". It is this kind of *inverse inference* that makes the "measurement model" useful—the logic is not that it rains because the friend has not come, but that only the fact that it rains ("state") can lead to the result that the friend has not come ("measurement")—In this exam-

ple, the measurement model is *deterministic*, in many real applications, we can only have a *probabilistic* measurement model. However, a measurement model, be it deterministic or probabilistic, can normally help our inference on the state.

With above introduced concepts **state**, **system model**, **measurement** and **measurement model**, we can now define estimation in a more formal way. Estimation of a state is a process of inferring the state out of measurements on the state, based on an available system model and an available measurement model. In many applications especially real-time applications, measurements will stream in from time to time; in these cases, we hope that whenever a new measurement arrives we can obtain a new state estimate based only on the old state estimate and the newly available measurement instead of re-performing an estimation based on all historical measurements. The practice of estimating a state recursively based only on the old state estimate and the newly available measurement in each time is called **recursive estimation**. Detailed mathematical formalism of recursive estimation from probabilistic perspective will be postponed to section 2.

This article is the first article of a planned series, i.e. the series "*A brief tutorial on recursive estimation with examples from intelligent vehicle applications*". One may treat this article as the "introduction" part of the series. In this article we would explain the basic spirit and utilities of recursive estimation in a general way, whereas in each of potential articles in future we would discuss one specific issue concerning (recursive) estimation in more details.

2 Recursive Estimation: Bayesian Inference

In recursive estimation for real-time applications, we denote the time index as t—Although the term "recursive" is not necessarily related to temporal iteration but reflects certain iteration in more general sense, we will still use the term "time index" throughout the presentation, on one hand for expression simplicity and on the other hand to show that recursive estimation is well rooted in real-time applications. We denote the state as x, denote the measurement as z; we use subscript t to denote the values of the state and the measurement at the instants specified by the subscript. For example, x_t and z_t denote respectively the state value and the measurement at time t; $x_{t_1:t_2}$ and $z_{t_1:t_2}$ denote respectively the state values and the measurements

from time t_1 to t_2.

We denote the system model as g:

$$x_t = g(x_{t-1}, u_t, \epsilon_t)$$

We denote the measurement model as h:

$$z_t = h(x_t, \gamma_t)$$

where u denotes the system input or the control signal; this part is not necessary for all applications and whether it exists depends on concrete applications. ϵ and γ denote respectively the system model noise or error and the measurement model noise or error.

In probabilistic terms, the system model and the measurement model can be represented compactly by two **conditional probability distributions** (CPD) $p(x_t|x_{t-1})$ (here we omit explicit representation of the system input term) and $p(z_t|x_t)$.

The estimation problem can be formulated as: given measurements $z_{1:t}$, compute the *a posteriori* distribution of x_t, i.e. compute $p(x_t|z_{1:t})$. Similarly, the recursive estimation problem can be formulated as: given the old estimate $p(x_{t-1}|z_{1:t-1})$ and the newly available measurement z_t, compute the new estimate $p(x_t|z_{1:t})$.

2.1 Bayesian inference

In this subsection, we review the generic methodology of solving the recursive estimation problem using Bayesian inference.

Before continuing, we review an important assumption in probability theory i.e. the **Markov assumption**: the future is independent of the past given the present. More specifically here, the Markov assumption means that $x_{(t+1,t+2,\ldots)}$ and $z_{(t+1,t+2,\ldots)}$ are independent of $x_{1:t-1}$ given x_t (for arbitrary time index t).

Based on the Markov assumption, we can carry out Bayesian inference as follows:

$$\begin{aligned} p(x_t|z_{1:t}) &= \frac{p(z_t|x_t, z_{1:t-1})p(x_t|z_{1:t-1})}{p(z_t|z_{1:t-1})} \\ &= \frac{p(z_t|x_t)p(x_t|z_{1:t-1})}{p(z_t|z_{1:t-1})} \end{aligned} \tag{1}$$

where $p(z_t|z_{1:t-1})$ is the normalization constant computed by:

$$p(z_t|z_{1:t-1}) = \int_{x_t} p(z_t|x_t, z_{1:t-1})p(x_t|z_{1:t-1})\,\mathrm{d}x_t$$

For discrete cases, the integral symbol $\int$ is replaced by the sum symbol $\sum$.

For the term $p(x_t|z_{1:t-1})$ in (1), we can expand it by applying the *law of total probability*:

$$\begin{aligned} p(x_t|z_{1:t-1}) &= \int_{x_{t-1}} p(x_t|x_{t-1}, z_{1:t-1})p(x_{t-1}|z_{1:t-1})\,\mathrm{d}x_{t-1} \\ &= \int_{x_{t-1}} p(x_t|x_{t-1})p(x_{t-1}|z_{1:t-1})\,\mathrm{d}x_{t-1} \end{aligned} \tag{2}$$

(2) together with (1) gives the mechanism of computing the new estimate $p(x_t|z_{1:t})$ recursively from the old estimate $p(x_{t-1}|z_{1:t-1})$ and the new measurement z_t. (2) is usually referred to as the **prediction** step, as it "predicts" an *a prior* distribution of current state x_t based on the old estimate. (1) is usually referred to as the **update** step, as it "updates" the state distribution to form an *a posterior* estimate by taking the newly available measurement z_t into account.

The prediction step specified in (2) and the update step specified in (1) are the generic two-steps methodology of recursive estimation using Bayesian inference. One may also refer to references such as [10] [11].

2.2 The Kalman filter

We have presented the generic methodology of recursive estimation using Bayesian inference. It seems that we have finished the story. On the contrary, we have just begun the story.

The formulas (2) and (1) are valuable mainly in theoretical sense, whereas they can hardly be applied directly in most applications. The reason lies in a forbidding computational burden and the impossibility to sample infinite potential values of the state.

We have to approximate the formalism (2) and (1) in certain way to make it tractable in real practice. A popular way is to approximate $p(x_t|x_{t-1})$ and $p(z_t|x_t)$ by linear-Gaussian models, which results in a famous recursive estimation method (family) i.e. the Kalman filter (KF) [12].

More specifically, let the system model g and the measurement model h be approximately represented by linear relationships:

$$x_t = \mathbf{A}x_{t-1} + \mathbf{B}u_t + \epsilon_t$$

$$z_t = \mathbf{H}x_t + \gamma_t$$

Assume $x_{t-1} \sim N(\hat{x}_{t-1}, \hat{\Sigma}_{x_{t-1}})$, $u_t \sim N(\hat{u}_t, \Sigma_u)$, $\epsilon_t \sim N(0, \Sigma_\epsilon)$, and $\gamma_t \sim N(0, \Sigma_\gamma)$, where N denotes the Normal or Gaussian distribution. Based on these assumptions, we can compute the *a priori* distribution (denoted as $\bar{x}_t$) and the *a posteriori* distribution (denoted as $\hat{x}_t$) of the state in the prediction step and the update step as follows.

Prediction: x_t (*a priori*) $\sim N(\bar{x}_t, \bar{\Sigma}_t)$
where

$$\bar{x}_t = \mathbf{A}\hat{x}_{t-1} + \mathbf{B}\hat{u}_t; \bar{\Sigma}_t = \mathbf{A}\hat{\Sigma}_{x_{t-1}}\mathbf{A}^T + \mathbf{B}\Sigma_u\mathbf{B}^T + \Sigma_\epsilon \tag{3}$$

Update: x_t (*a posteriori*) $\sim N(\hat{x}_t, \hat{\Sigma}_t)$
where

$$\hat{x}_t = \bar{x}_t + \mathbf{K}(z_t - \mathbf{H}\bar{x}_t); \hat{\Sigma}_t = (\mathbf{I} - \mathbf{K}\mathbf{H})\bar{\Sigma}_t \tag{4}$$
$$\text{where } \mathbf{K} = \bar{\Sigma}_t\mathbf{H}^T(\mathbf{H}\bar{\Sigma}_t\mathbf{H}^T + \Sigma_\gamma)^{-1}$$

Formulas (3) and (4) are a commonly encountered formalism of the Kalman filter in literature. Detailed derivation of (3) and (4) by applying above linear-Gaussian assumptions to (1) and (2) is omitted here. Instead, we would explain the Kalman filter more from another perspective (the "information" perspective) to clarify the essence of this famous estimation method.

Given $\{x_1, \Sigma_1\}$ and $\{x_2, \Sigma_2\}$ are two source estimates of a state x; we want to obtain a new estimate (called "fusion estimate") of x out these two source estimates. The covariances Σ_1 and Σ_2 reflect the uncertainty of the two estimates, thus we can use the inverses of the covariances i.e. Σ_1^{-1} and Σ_2^{-1} to indicate the "quality" of the estimates: the smaller the covariance of an estimate is, the higher the quality of the estimate is. This "quality" is usually referred to more formally as "information" (hence the covariance inverse is referred to as "information matrix").

By so far, a rather intuitive idea of how to fuse the two estimates may come into our mind, i.e. forming a weighted combination of the two estimates and giving a weight to each estimate proportional to its "quality". This intuitive idea can be formulated as:

$$\hat{x} = (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}(\Sigma_1^{-1}x_1 + \Sigma_2^{-1}x_2); \hat{\Sigma} = (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1} \tag{5}$$

Why introduce this intuitive idea of fusing two estimates? The answer is simple, because this idea, formulated in (5), is equivalent to the Kalman filter. To better understand this point, we can reformulate (5) as follows:

$$\begin{aligned} \mathbf{K} &= \Sigma_1(\Sigma_1 + \Sigma_2)^{-1} \\ \hat{x} &= x_1 + \mathbf{K}(x_2 - x_1); \hat{\Sigma} = (\mathbf{I} - \mathbf{K})\Sigma_1 \end{aligned} \tag{6}$$

Now one can easily see the similarity between (6) and (4). (5) and (6) are for cases where both estimates are full-rank observations. Given a case where one estimate (say x_2) has a rank-deficient observation matrix $\mathbf{H}$ i.e. $x_2 = \mathbf{H}x$, one can derive from (5) or (6) a formalism essentially the same to (4) i.e.:

$$\hat{x} = x_1 + \mathbf{K}(x_2 - \mathbf{H}x_1); \hat{\Sigma} = (\mathbf{I} - \mathbf{KH})\Sigma_1 \tag{7}$$

$$\text{where } \mathbf{K} = \Sigma_1\mathbf{H}^T(\mathbf{H}\Sigma_1\mathbf{H}^T + \Sigma_2)^{-1}$$

One can refer to [2] for the derivation (if P_{1d} and P_{2d} in (6) in [2] are set zero). One can easily verify that the formalism (7) always holds regardless of whether the observation matrix $\mathbf{H}$ has full rank or is rank deficient.

From above interpretation, one can see that the Kalman filter, in essence, is a fusion method that forms the fusion estimate by a **linear weighted combination** of source estimates.

The Kalman filter formalism (3) and (4) apply directly to linear systems. For non-linear systems, one may carry out an adapted version of the Kalman filter by linearizing the system model and the measurement model locally around the old estimate and the *a priori* estimate and hence obtain the extended Kalman filter (EKF) [13].

2.3 The particle filter

The Kalman filter (including its variants such as the EKF) is generally an effective and efficient method for unimodal estimation problems. In some applications, the state distribution may be multimodal—this is likely to happen especially when measurement ambiguity exists i.e. there are multiple measurements on a state at the same time and one can not be sure which measurement actually correspond to the state—The need to handle multimodal ambiguities lead to another well known recursive estimation method, the particle filter (or a sequential Monte Carlo method) [14] [15].

The basic idea is to approximate the state distribution $p(x_t|z_{1:t})$ directly by a group of finite state samples called **particles** $\{x_t^j|j = 1, ..., N\}$ together

with corresponding **weights** $\{w_t^j | j = 1, ..., N\}$ and approximate the evolution of the state by a sampling process $x_t \sim q(x_t|x_{t-1}, z_t)$, as follows:
Prediction:

$$\bar{x}_t^j \sim q(x_t|x_{t-1}^j, z_t);$$

Update:

$$\begin{aligned} x_t^j &= \bar{x}_t^j; \\ \bar{w}_t^j &= \frac{w_{t-1}^j p(z_t|\bar{x}_t^j) p(\bar{x}_t^j|x_{t-1}^j)}{q(\bar{x}_t^j|x_{t-1}^j, z_t)} \\ w_t^j &= \bar{w}_t^j / \sum_{j=1}^{N} \bar{w}_t^j \end{aligned}$$

where $q(x_t|x_{t-1}, z_t)$ is called the **proposal distribution (function)**. Often, it is chosen the same to $p(x_t|x_{t-1})$ and the weight update step is reduced to a simple form:

$$\bar{w}_t^j = w_{t-1}^j p(z_t|\bar{x}_t^j)$$

As one can easily see the simplicity of this reduced form, one may naturally pose the question: why do we bother to keep the proposal distribution term in the formalism of the particle filter? Reasons are two-folds. First, the $p(x_t|x_{t-1})$ may be complicated and it may be difficult to generate samples according to $p(x_t|x_{t-1})$. In this case, we would prefer a tractable proposal distribution according to which samples can be generated more easily. Second, using $p(x_t|x_{t-1})$ as the proposal distribution may not work well in cases where the distribution $p(z_t|x_t)$ (representing the measurement model) is much narrower than the distribution $p(x_t|x_{t-1})$ (representing the system model), in other words, in cases where the sensor measurements are more informative than the *a priori* predicted by the system model. In these cases, most of the sample particles generated according to the system dynamics will get very low weight in the update step and hence waste the sampling resource. To handle these cases, we would prefer a proposal distribution that can concentrate the sample particles more around measurements.

As the particle filtering process continues, the particles will gradually diverge and many of them will drift to states with low weight updating ratio. This is a severe waste of the sampling resource, which quickly degrades the performance of the particle filter. To handle this problem, a common technique is usually employed, i.e. the **resampling** technique: if the weights

$w_t^j | j = 1, ..., N$ are not distributed "uniformly" enough (For example, the indicator to the uniform degree of the distribution of the weights can be chosen as the inverse of the summed squares of the weights), a heuristic step called *resampling* will take place, which draw new particles from current particles with their chance to be selected proportional to their associated weights and reset all weights to $1/N$.

3 Examples: Vehicle Localization

In this subsection, we demonstrate the utilities of recursive estimation with examples from intelligent vehicle applications. More specifically, we examine the cases of vehicle localization and demonstrate the performance of the Kalman filter.

3.1 Vehicle localization in a 1D case

3.1.1 Application description

Suppose a vehicle is moving on a straight road and we treat only its longitudinal position as its state denoted as p_x. The system model is given as the following kinematic model:

$$p_{x,t} = p_{x,t-1} + v_{x,t}\Delta T \tag{8}$$

where v_x denotes the vehicle longitudinal speed which is regarded as the system input; ΔT denotes the system period. Suppose the vehicle is equipped with a speedometer that monitors the vehicle speed. The speedometer outputs, denoted as $\hat{v}_x$, suffer from certain errors which are assumed to follow the Gaussian distribution with zero mean and the covariance Σ_v, i.e. $v_{x,t} \sim N(\hat{v}_{x,t}, \Sigma_v)$.

Suppose the vehicle is also equipped with a GPS (Global Positioning System) that outputs directly measurements on the vehicle longitudinal position i.e. the vehicle state. Let GPS measurements be denoted as z_x. Then the measurement model is given as:

$$z_{x,t} = p_{x,t} + \gamma_t \tag{9}$$

where γ denotes the measurement error which is assumed to follow the Gaussian distribution with zero mean and the covariance Σ_γ, i.e. $\gamma \sim N(0, \Sigma_\gamma)$.

Carry out the prediction and update steps (3) and (4) in the Kalman filter. Note that the system model error is zero here.
Prediction:

$$\bar{p}_{x,t} = \hat{p}_{x,t-1} + \hat{v}_{x,t}\Delta T$$
$$\bar{\Sigma}_{p_x,t} = \hat{\Sigma}_{p_x,t-1} + \Sigma_v \Delta T^2$$

Update:

$$\mathbf{K} = \bar{\Sigma}_{p_x,t}(\bar{\Sigma}_{p_x,t} + \Sigma_\gamma)^{-1}$$
$$\hat{p}_{x,t} = \bar{p}_{x,t} + \mathbf{K}(z_{x,t} - \bar{p}_{x,t})$$
$$\hat{\Sigma}_{p_x,t} = (\mathbf{I} - \mathbf{K})\bar{\Sigma}_{p_x,t}$$

3.1.2 Simulation

We test the performance of the Kalman filter using synthetic data generated according to the system model (8) and the measurement model (9). In the simulation, let $\Delta T = 1(s)$; let $\Sigma_v = 0.5^2(m^2/s^2)$; let $\Sigma_\gamma = 10^2(m^2)$. Set the ground-truth $p_{x,0} = 0(m)$ and $v_{x,t} = 10(m/s)$. The speedometer outputs are synthesized according to $\hat{v}_{x,t} \sim N(v_{x,t}, \Sigma_v)$ and the GPS measurements are synthesized according to $z_{x,t} \sim N(p_{x,t}, \Sigma_\gamma)$.

The Kalman filter was applied to the synthetic data and estimates on the vehicle state (i.e. the vehicle longitudinal position) were obtained. The estimate errors were computed and compared with the measurement errors. The result of one trial is shown in Fig.1. For a statistical evaluation, 100 Monte Carlo trials were performed and the result is shown in Fig.2. As we can see, the estimate errors are apparently smaller than the measurement errors—This demonstrates well one merit of estimation mentioned previously in section 1 i.e. a proper estimation method can provide more precise estimates of a state than raw measurements do. This is also why many estimation methods are called "filters" such as the Kalman filter, particle filter etc, because they do "filter" measurement noises.

3.2 Vehicle localization in a 2D case

3.2.1 Application description

Now consider a more general case where the vehicle is navigating on a 2D plane. In this case, vehicle localization consists in estimating the pose (the

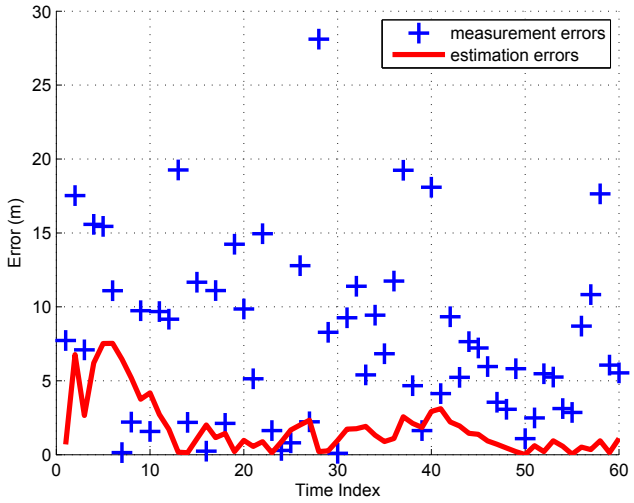

Figure 1: Estimation and measurement errors for one trial

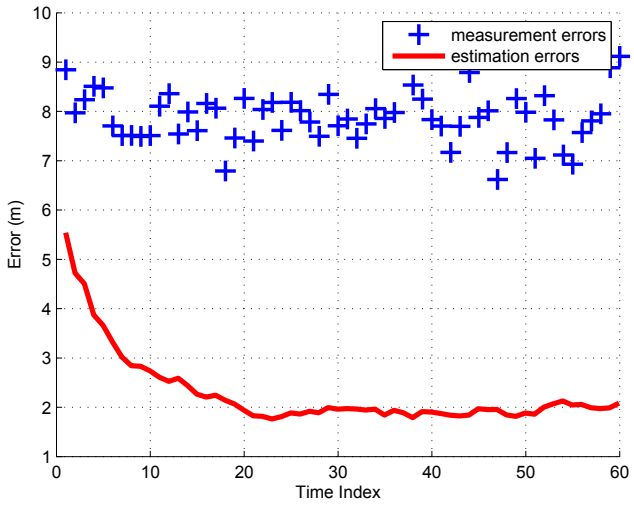

Figure 2: Estimation and measurement errors for 100 Monte Carlo trials

position (x, y) as well as the orientation θ) of the vehicle. In other words,

we treat the pose of the vehicle as its state and try to estimate this state denoted compact as $\mathbf{p}$ i.e. $\mathbf{p} = (x, y, \theta)$. The system model is given as the following kinematic model:

$$\begin{cases} x_t &= x_{t-1} + v_t \Delta T cos(\theta_{t-1} + w_t \Delta T/2) \\ y_t &= y_{t-1} + v_t \Delta T sin(\theta_{t-1} + w_t \Delta T/2) \\ \theta_t &= \theta_{t-1} + w_t \Delta T \end{cases} \tag{10}$$

where ΔT denotes the system period; v and w denote respectively the speed and the yawrate of the vehicle. Suppose the vehicle is equipped with devices that monitor its speed and its yawrate. The speed measurements are denoted as $\hat{v}$, and yawrate measurements are denoted as $\hat{w}$. Their errors are assumed to follow the Gaussian distribution as $\Delta v_t \sim N(0, \Sigma_v)$ and $\Delta w_t \sim N(0, \Sigma_w)$.

Suppose the vehicle is also equipped with a GPS that outputs measurements on the vehicle position (x, y). Let GPS measurements be denoted as $\mathbf{z}$ and the measurement model is given as:

$$\mathbf{z}_t = \mathbf{H}\mathbf{p}_t + \gamma_t \tag{11}$$

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

where γ denotes the measurement error which is assumed to follow the Gaussian distribution with zero mean and the covariance Σ_γ, i.e. $\gamma \sim N(\mathbf{0}, \Sigma_\gamma)$. As we can see, the measurement model given in (11) is a partial measurement model. We do not have direct measurements on the orientation part of the state and we can only reveal the orientation part indirectly through proper estimation.

The system model (10) is a nonlinear model with respect to the vehicle orientation θ. We have linearize locally the system model in order to apply the formulas (3) and (4). This adapted Kalman filter with local linearization is called *extended Kalman filter* (EKF)—It is worth noting that "local linearization" can refer not only to local linearization of the system model but also to that of the measurement model if the measurement model is nonlinear. In the example presented here, the measurement model is already linear and hence exempt from a preliminary step of local linearization.

The locally linearized system model is rewritten as follows:

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} \approx \begin{bmatrix} \bar{x}_t \\ \bar{y}_t \\ \bar{\theta}_t \end{bmatrix} + \mathbf{A}(\mathbf{p}_{t-1}, \mathbf{u}_t) \begin{bmatrix} \Delta x_{t-1} \\ \Delta y_{t-1} \\ \Delta \theta_{t-1} \end{bmatrix} + \mathbf{B}(\mathbf{p}_{t-1}, \mathbf{u}_t) \begin{bmatrix} \Delta v_t \\ \Delta w_t \end{bmatrix} \tag{12}$$

$$\begin{cases} \bar{x}_t &= x_{t-1} + v_t \Delta T cos(\theta_{t-1} + w_t \Delta T/2) \\ \bar{y}_t &= y_{t-1} + v_t \Delta T sin(\theta_{t-1} + w_t \Delta T/2) \\ \bar{\theta}_t &= \theta_{t-1} + w_t \Delta T \end{cases}$$

$$\mathbf{A}(\mathbf{p}_{t-1}, \mathbf{u}_t) = \begin{bmatrix} 1 & 0 & -v_t \Delta T sin(\theta_{t-1} + w_t \Delta T/2) \\ 0 & 1 & v_t \Delta T cos(\theta_{t-1} + w_t \Delta T/2) \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{B}(\mathbf{p}_{t-1}, \mathbf{u}_t) = \begin{bmatrix} \Delta T cos(\theta_{t-1} + w_t \Delta T/2) & -v_t \Delta T^2 sin(\theta_{t-1} + w_t \Delta T/2)/2 \\ \Delta T sin(\theta_{t-1} + w_t \Delta T/2) & v_t \Delta T^2 cos(\theta_{t-1} + w_t \Delta T/2)/2 \\ 0 & \Delta T \end{bmatrix}$$

where $\mathbf{u}_t = (v_t, w_t)$. As we can see, the matrices $\mathbf{A}(\mathbf{p}_{t-1}, \mathbf{u}_t)$ and $\mathbf{B}(\mathbf{p}_{t-1}, \mathbf{u}_t)$ are actually the Jacobian matrices of the state evolution function (specified in (10)) with respect to $\mathbf{p}_{t-1}$ and $\mathbf{u}_t$ respectively. With this locally linearized system model (12) and the measurement model (11), we can carry out the prediction and update steps (3) and (4) in the Kalman filter—It is worth noting that there exists linearization error and this error may be modeled in certain way and treated as the system model error. Since this linearization error is not essential to demonstrating the mechanism of the adapted Kalman filter with local linearization i.e. the EKF, we neglect it here.

Prediction:

$$\begin{bmatrix} \bar{x}_t \\ \bar{y}_t \\ \bar{\theta}_t \end{bmatrix} = \begin{bmatrix} \hat{x}_{t-1} + \hat{v}_t \Delta T cos(\hat{\theta}_{t-1} + \hat{w}_t \Delta T/2) \\ \hat{y}_{t-1} + \hat{v}_t \Delta T sin(\hat{\theta}_{t-1} + \hat{w}_t \Delta T/2) \\ \hat{\theta}_{t-1} + \hat{w}_t \Delta T \end{bmatrix}$$

$$\bar{\Sigma}_{\mathbf{p},t} = \mathbf{A}(\hat{\mathbf{p}}_{t-1}, \hat{\mathbf{u}}_t) \hat{\Sigma}_{\mathbf{p},t-1} \mathbf{A}(\hat{\mathbf{p}}_{t-1}, \hat{\mathbf{u}}_t)^T + \mathbf{B}(\hat{\mathbf{p}}_{t-1}, \hat{\mathbf{u}}_t) \begin{bmatrix} \Sigma_v & 0 \\ 0 & \Sigma_w \end{bmatrix} \mathbf{B}(\hat{\mathbf{p}}_{t-1}, \hat{\mathbf{u}}_t)^T$$

Update:

$$\mathbf{K} = \bar{\Sigma}_{\mathbf{p},t} \mathbf{H}^T (\mathbf{H} \bar{\Sigma}_{\mathbf{p},t} \mathbf{H} + \Sigma_\gamma)^{-1}$$

$$\hat{\mathbf{p}}_t = \begin{bmatrix} \bar{x}_t \\ \bar{y}_t \\ \bar{\theta}_t \end{bmatrix} + \mathbf{K}(z_t - \mathbf{H} \begin{bmatrix} \bar{x}_t \\ \bar{y}_t \\ \bar{\theta}_t \end{bmatrix})$$

$$\hat{\Sigma}_{\mathbf{p}_x,t} = (\mathbf{I} - \mathbf{K}\mathbf{H}) \bar{\Sigma}_{\mathbf{p},t}$$

3.2.2 Simulation

We test the performance of the Kalman filter using synthetic data that generated according to the system model (10) and the measurement model

(11). In the simulation, let $\Delta T = 1(s)$; let $\Sigma_v = 0.5^2(m^2/s^2)$; let $\Sigma_w = 0.02^2(rad^2/s^2)$; let $\Sigma_\gamma = diag(10^2, 10^2)(m^2)$. Set the ground-truth $p_0 = [0(m), 0(m), -\pi/2(rad)]^T$; $v_t = 10(m/s)$ and $w_t = 0.04(rad/s)$. The speed measurements and the yawrate measurements are synthesized according to $\hat{v}_t \sim N(v_t, \Sigma_v)$ and $\hat{w}_t \sim N(w_t, \Sigma_w)$. The GPS measurements are synthesized according to $z_t \sim N(p_t, \Sigma_\gamma)$.

The Kalman filter was applied to the synthetic data and estimates on the vehicle state (i.e. the vehicle position as well as the vehicle orientation) were obtained. The result of one trial is shown in Fig.3, in which the black line represents the ground-truth of the vehicle trajectory, the red line represents the estimated vehicle trajectory, and the blue crosses represent GPS measurements on the vehicle position. As we can see, the estimated vehicle trajectory is noticeably smoother than the jumping measurements.

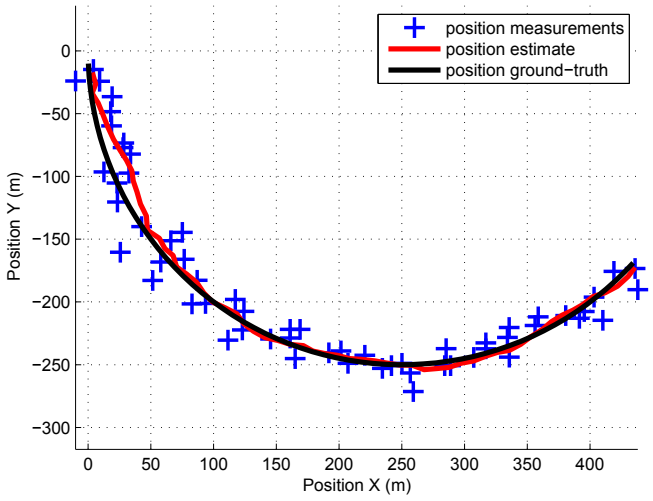

Figure 3: Position measurements and estimates for one trial

The position estimate errors were computed and compared with the position measurement errors. The result of 100 Monte Carlo trials is shown in Fig.4. As we can see, the estimate errors are also apparently smaller than the measurement errors, similar to the results in previous example.

The orientation estimate errors were also computed and the result of the same 100 Monte Carlo trials is shown in Fig.5. As we can see, the orientation

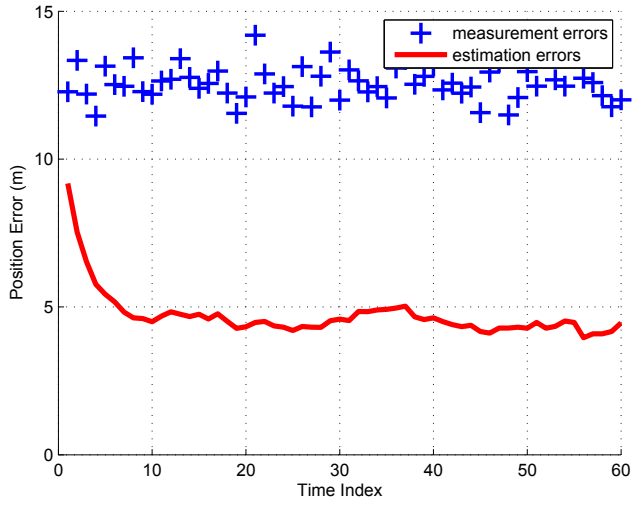

Figure 4: Position estimate and measurement errors for 100 Monte Carlo trials

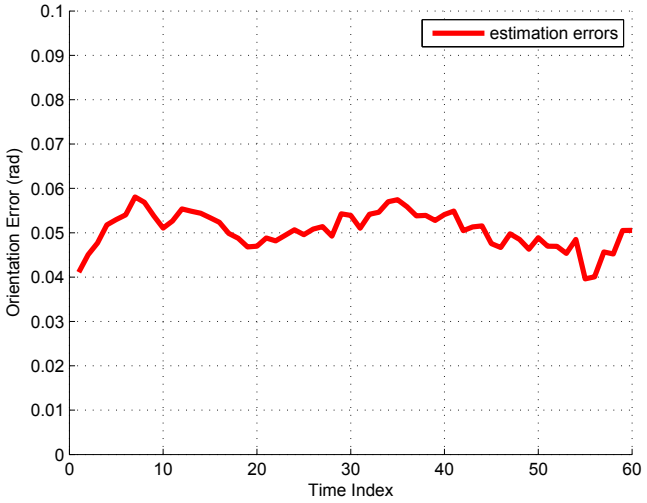

Figure 5: Orientation estimate errors for 100 Monte Carlo trials

estimate errors vary around 0.05 rad i.e. a bit smaller than 3 degrees (note that the tiny angle formed by two neighboring tick marks on a clock hold even 6 degrees). Here, we do not intend to examine isolated the quality of the orientation estimates, because the quality of these estimates including all above demonstrated results many change given different vehicle configurations. What we try to highlight is: without estimation in this application, we can not have any estimate of the vehicle orientation except a random guess—concerning the orientation, the error of a random guess can be as large as $\pi/2$ i.e. 180 degrees—This example demonstrates well another important utility of estimation i.e. a proper estimation method can reveal state part that is not directly measurable.

It is worth noting that not all kinds of states can be either directly or indirectly observable. The **observability** of a state depends on the system model as well as the measurement model. On can refer to control theory literature such as [16] for more explanations on this issue.

4 Conclusion

In this brief tutorial, we have reviewed several fundamental concepts concerning estimation, i.e. state, system model, measurement, and measurement model. We have reviewed the generic formalism of recursive estimation using Bayesian inference and have reviewed concrete formulas of two commonly used recursive estimation methods i.e. the Kalman filter and the particle particle. For the Kalman filter, we have explained its essence from "information" perspective and presented two examples of its application to vehicle localization problems. With these examples, we have demonstrated two important utilities of estimation, summarized in short words, i.e. "can know" and "know better".

This brief tutorial is far away from and by no means intended to be a comprehensive survey of estimation methods. It is only intended to enlighten beginners on basic spirit of recursive estimation and how recursive estimation can potentially benefit concrete applications and to arouse their interests in studying more profound issues on estimation such as estimation methods for handling cooperative systems [2] [17] [18].

References

[1] H. Li, F. Nashashibi, and G. Toulminet. Localization for intelligent vehicle by fusing mono-camera, low-cost gps and map data. In *IEEE International Conference on Intelligent Transportation Systems*, pages 1657–1662, 2010.

[2] H. Li, F. Nashashibi, and M. Yang. Split covariance intersection filter: Theory and its application to vehicle localization. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1860–1871, 2013.

[3] H. Li and F. Nashashibi. Cooperative multi-vehicle localization using split covariance intersection filter. *IEEE Intelligent Transportation Systems Magazine*, 5(2):33–44, 2013.

[4] H. Li and F. Nashashibi. Multi-vehicle cooperative localization using indirect vehicle-to-vehicle relative pose estimation. In *IEEE International Conference on Vehicular Electronics and Safety*, pages 267–272, 2012.

[5] H. Li and F. Nashashibi. Robust real-time lane detection based on lane mark segment features and general a priori knowledge. In *IEEE International Conference on Robotics and Biomimetics*, pages 812–817, 2011.

[6] H. Li and F. Nashashibi. Lane detection (part i): Mono-vision based method. *INRIA Tech Report*, RT-433, 2013.

[7] H. Li, F. Nashashibi, B. Lefaudeux, and E. Pollard. Track-to-track fusion using split covariance intersection filter-information matrix filter (scif-imf) for vehicle surrounding environment perception. In *IEEE International Conference on Intelligent Transportation Systems*, pages 1430–1435, 2013.

[8] H. Li and F. Nashashibi. Multi-vehicle cooperative perception and augmented reality for driver assistance: A possibility to see through front vehicle. In *IEEE International Conference on Intelligent Transportation Systems*, pages 242–247, 2011.

[9] H. Li, M. Tsukada, F. Nashashibi, and M. Parent. Multivehicle cooperative local mapping: A methodology based on occupancy grid map

merging. *IEEE Transactions on Intelligent Transportation Systems*, in press, 2014.

[10] K.P. Murphy. *Dynamic Bayesian networks: Representation, inference and learning*. Ph.D. Dissertation, UC Berkeley, 2002.

[11] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT Press, 2005.

[12] R.E. Kalman. A new approach to linear filtering and prediction problem. *ASME Trans, Ser. D, J. Basic Eng.*, 82:35–45, 1960.

[13] M.S. Grewal and A.P. Andrews. *Kalman filtering: Theory and practice*. New York, USA: Wiley, 2000.

[14] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[15] A. Doucet, N. De Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. New York, USA: Springer-Verlag, 2001.

[16] E. Sontag. *Mathematical control theory: Deterministic finite dimensional systems*. Springer, 1998.

[17] S.J. Julier and J.K. Uhlmann. General decentralized data fusion with covariance intersection (ci). *Handbook of Data Fusion*, 2001.

[18] H. Li. *Cooperative perception: Application in the context of outdoor intelligent vehicle systems*. Ph.D. Dissertation, Mines ParisTech, 2012.