Guidance Navigation and Control
Digested informal lecture notes on state estimation

October 10, 2019

1 Formal representation of the states of our system

We represent the states of our system with a vector q containing as elements *probability density functions* (or *pdf* for short) instead of containing real numbers. We employ such representation because the *pdf* contains information about the uncertainty that we might have on a property of the system, whereas a simple real number cannot encode such information. Therefore, we represent the states of our system with the column vector $q = \begin{bmatrix} f_1 & f_2 & \dots & f_n \end{bmatrix}^T$, where n is the number of states of the system, and f_n are the different *pdfs* (to be defined shortly) associated to each of our states.

Let us define the *pdf* as $f(\cdot) := \mathcal{V} \to \mathbb{R}_+$, where $\mathcal{V}$ is the space where state of interest belongs to, for example, a position $p_x \in \mathbb{R}$ in meters or temperature $\theta \in \mathbb{R}_+$ in Kelvins, and the output of $f(\cdot)$ is always positive. The *pdf* is giving us the information about the probability of a random variable to be between two values. Therefore the probability of having the random variable between $-\infty$ and ∞ is 1 (or informally 100%). We can write this fact formally as follows

$$\text{Prob}[x \in (-\infty, \infty)] := \int_{-\infty}^{\infty} f(x)dx = 1. \tag{1}$$

Note that $\text{Prob}[x \in (a, a)] = 0$, $a \in \mathbb{R}$ if $f(x)$ is *smooth enough*.

There exist many different *pdf* depending on the nature of the variable and its uncertainty, such as *uniform*, *normal*, *exponential*, etc. You can click on this link to find information about them `https://www.statlect.com/probability-distributions/`.

We are interested in two properties of the *pdf*. The first one is the expected value, or $E[x] = \hat{x}$, where

$$E[x] = \int_{-\infty}^{\infty} f(x)\, x\, dx, \tag{2}$$

where if $f(x)$ is symmetric, then $\hat{x}$ is the center point of such symmetry.

The second property is the *variance*, which is a particular case of the *covariance* $CoVar[x, y] := E[(x - \hat{x})(y - \hat{y})]$. The variance is calculated when we set $x = y$, then it measures the expected squared *distance* of your random variable to its expected value. This is also known as σ_x^2, and note that obviously is a non-negative number. In fact, σ_x^2 is a positive number for a generic distribution. If you consider $\sigma_x^2 = 0$, then it must be true that $\text{Prob}[x \in (\hat{x}, \hat{x})] = 1$, which does not hold in general, at least not for the *uniform* or the *normal pdf*.

1.1 Simple fusion algorithm

If you have n *pdf* from $n \in \mathbb{N}$ **uncorrelated sources** referring to the same state, then you can fuse them as follows

$$\text{Prob}[x \in (-\infty, \infty)] = \frac{1}{\alpha} \int_{-\infty}^{\infty} \prod_n f_n(x)\, dx = 1, \tag{3}$$

where $\alpha \in \mathbb{R}_+$ is a positive scaling factor to calculate such that we have 1 as a total probability. Therefore, if we would like to calculate the probability of finding $x \in (a_1, a_2)$, then

$$\text{Prob}[x \in (a_1, a_2)] = \frac{1}{\alpha} \int_{a_1}^{a_2} \prod_n f_n(x)\, dx. \tag{4}$$

Note that the expression (4) is valid for any generic *pdf*. Furthermore, they can be different in nature like f_1 is *uniform* and f_2 *normal*.

Of course, if you want to find out the expected value after the fusion, you must apply the definition in (2)

$$E[x] = \int_{-\infty}^{\infty} \frac{1}{a} \prod_n f_n(x)\, x\, dx, \tag{5}$$

and the same for the variance. These two operations can be very demanding numerically (computational power). That will motivate the study of the following *normal probability density function*.

1.2 *Normal pdf*

The *normal pdf* is a popular function that approximates very well many natural phenomena, and we are lucky that it is tractable mathematically. Specially when we are *fusioning* different normal *pdf*, and when these normal *pdf* goes through linear transformations, e.g., a linear dynamics for a model.

Let us define the normal *pdf* as

$$g(x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} \exp\left[\frac{(x - \hat{x})^2}{2\sigma_x^2}\right], \tag{6}$$

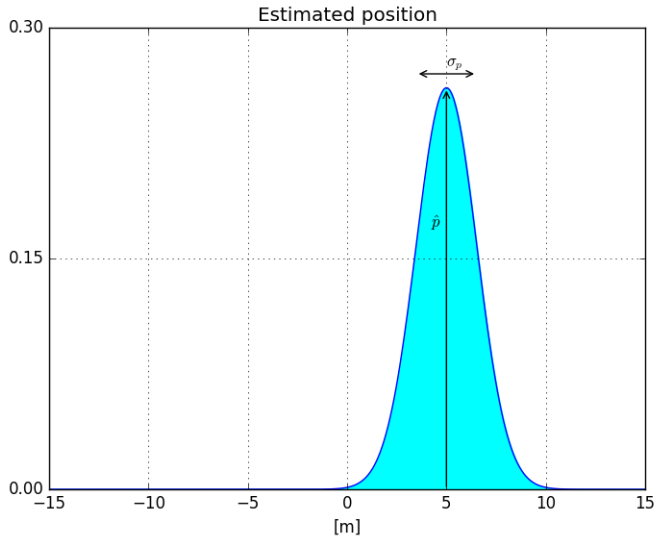

Figure 1: Normal probability density function, with $\hat{p}$ being the expected position of a vehicle in the x axis, and σ_p being the standard deviation calculated from the variance σ_p^2. The range $\hat{p} \pm \sigma_p$ includes the 68% of all the values in the distribution approximately.

where we plot it in Figure 1. Note that $g(x)$ is characterized uniquely by two parameters, the expected value $\hat{x}$ and the variance σ_x^2.

1.3 Sensor fusion of two normal *pdf*

According to (3) we can fuse two normal *pdf*, if they come from independent sources, as follows

$$g_f(x) = g_1(x)g_2(x), \tag{7}$$

which is an easy operation since we are dealing with exponentials. In particular, we have that $g_f(x)$ is characterized by

$$\begin{cases} \hat{x}_f &= \frac{\hat{x}_1\sigma_{x_2}^2 + \hat{x}_2\sigma_{x_1}^2}{\sigma_{x_1}^2 + \sigma_{x_2}^2} \\ \sigma_f^2 &= \frac{\sigma_{x_1}^2\sigma_{x_2}^2}{\sigma_{x_1}^2 + \sigma_{x_2}^2} \end{cases}. \tag{8}$$

Note that we just derived the Kalman gain from the famous Kalman filter. If we define $k := \frac{\sigma_{x_1}^2}{\sigma_{x_1}^2 + \sigma_{x_2}^2}$ then we can rewrite (8) as

$$\begin{cases} \hat{x}_f &= \hat{x}_1 + k(\hat{x}_2 - \hat{x}_1) \\ \sigma_f^2 &= \sigma_1^2 - k\sigma_1^2 \end{cases}. \tag{9}$$

The normal *pdf* is a particular case of a *Pearson distribution* `https://en.wikipedia.org/wiki/Pearson_distribution`, and expressions (or nice approximations) like (8) can be found for other distributions as well.

2 Modeling the dynamics of our states. Linear systems and normal *pdfs*

If we represent our states with real numbers, it is common to find the equations that predict the evolution of our states at different fixed time intervals (discrete time) as

$$q(k+1) = f(q(k), u(k)), \tag{10}$$

where $k \in \mathbb{N}$ is the time event, $q \in \mathbb{R}^n$ (with $n \in \mathbb{N}$ being the number of states), $u \in \mathbb{R}^p$ (with $n \in \mathbb{N}$ being the number of inputs), and $f := \mathbb{R}^n \times \mathbb{R}^p \to \mathbb{R}^n$ being the *state-transition* function. If we can write (10) as a linear system, i.e.,

$$q(k+1) = Fq(k) + Gu(k), \tag{11}$$

with $F \in \mathbb{R}^{n \times n}$ and $G \in \mathbb{R}^{n \times p}$, then we have very good news when the elements of q are not real numbers but normal *pdfs*. In particular, linear combinations (as in (11)) of normal *pdfs* have as a result another normal *pdf*, check this link for the proof `https://www.statlect.com/probability-distributions/normal-distribution-linear-combinations`. **That is not the case in general for arbitrary *pdf*s.** One nice consequence of this fact is that if we take a normal distribution g_f that is a linear combination of other normal distributions g_n, then the expected value of g_f is also the same linear combination of the expected values from g_n, i.e,

$$\hat{q}(k+1) = F\hat{q}(k) + G\hat{u}(k), \tag{12}$$

so we can reuse F and G from (11)!

Remember that a normal distribution is characterized by its expected value $\hat{x}$ and variance σ_x^2. Then, together with (12) we need to find out the evolution of the variances of the considered normal distributions. Consider the following *covariance* matrix for your vectorial state q

$$P := CoVar(q,q) := \begin{bmatrix} CoVar(q_1,q_1) & \dots & CoVar(q_1,q_n) \\ \vdots & \ddots & \vdots \\ CoVar(q_n,q_1) & \dots & CoVar(q_n,q_n) \end{bmatrix} = E[(q-\hat{q})(q-\hat{q})^T]. \tag{13}$$

Note that by definition we have that $\sigma_{q_n}^2 = CoVar(q_n, q_n)$, so the diagonal elements are the covariances of the states in q.

Now we check the following identity for the vector q and a matrix F.

$$\begin{aligned} CoVar(Fq, Fq) &= E[(Fq - F\hat{q})(Fq - F\hat{q})^T] \\ &= E[F(q - \hat{q})(q - \hat{q})^T F^T] \\ &= FE[(q - \hat{q})(q - \hat{q})^T]F^T \\ &= F\,CoVar(q, q)\,F^T = FPF^T \end{aligned} \tag{14}$$

Note that F is a constant matrix, and all constants can get out of the integral (2) for calculating expected values. That is why the last step in (14).

Now let us calculate

$$CoVar(q(k+1), q(k+1)) = CoVar(Fq(k) + Gu(k), Fq(k) + Gu(k)). \tag{15}$$

From the definition we can check that the covariance of the linear combination of two variables is given by

$$CoVar(ax+by) = a^2\,CoVar(x, x)+b^2\,CoVar(y, y)+2ab\,CoVar(x, y). \tag{16}$$

The covariance $CoVar(x, y) = 0$ means that x and y are not correlated, i.e., if we assume that $q(k)$ and $u(k)$ are not correlated (state and input at the same time instant are independent of each other), then we can continue (15) as

$$\begin{aligned} P(k+1) &= CoVar(Fq(k) + Gu(k), Fq(k) + Gu(k)) \\ &= CoVar(Fq(k), Fq(k)) + CoVar(Gu(k), Gu(k)) \\ &= FP(k)F^T + G\,CoVar(u(k), u(k))\,G^T. \\ &= FP(k)F^T + GQ(k)G^T \end{aligned} \tag{17}$$

Note that Q in (17) is a diagonal matrix if there are no correlations between the inputs. The diagonal elements of Q are just $\sigma^2_{u_i}, i \in \{1, \ldots, p\}$. Therefore, we can predict the evolution in (discrete) time of a normal distribution by iterating (11) for the expected value of q and (17) for the variance of q.

3 Measurements: Inputs & observations in linear systems

Equations (12) and (17) indicate how the *pdf*'s associated to the states q evolve **if** the *pdf*'s are normal, and the process that we consider can be modelled as a linear system. Therefore, we can make predictions over time starting from an initial *pdf*. In particular from the initials $\hat{q}(0)$ and $P(0)$.

We can employ measurements for both, to make predictions and to improve/update the information that carries our *pdf*. If the measurements

are employed to predict in (10), then they are considered *inputs*. If the measurements are compared with a prediction, then they are considered as *observations*.

If we only consider inputs, then the uncertainty of the input u will increase the uncertainty of our states in q each time we iterate (17). The observations can be employed to improve our estimation of the *pdfs* associated with q by following *sensor fusion* as in (8) or (9). When we fuse two sources of information, we must ensure they represent the same quantity, for example, they are both velocities and they are in the same units.

Example of observation: We might measure a velocity employing a pressure sensor, which is proportional to the square of the velocity. Consider, $q = \begin{bmatrix} p_x & v_x & b_a \end{bmatrix}^T$, where p_x is a position, v_x is a velocity and b_a is a bias from an accelerometer. Then the *observation function* that gives the squared velocity will be

$$y = h(q) = v_x^2 \neq Hq. \tag{18}$$

We call $y \in \mathbb{R}$ the *predicted observation* (it might be a vector of course), which will be fused with the actual measurement later on. However, we note that (18) cannot be written as a function of a matrix $H \in \mathbb{R}^{1\times 3}$ multiplying the state q. Therefore, (18) is said to be *nonlinear*. We are interested in *linear observation functions* since we know that the following holds for normal *pdf*'s.

$$\hat{y} = H\hat{q}, \tag{19}$$

i.e., the predicted observation is also a normal *pdf*, and its associated covariance

$$P_y = CoVar(Hq, Hq) = HPH^T. \tag{20}$$

The actual measurement must be also associated to a *pdf*, therefore it will be described by the corresponding $\hat{y}_m$ and P_{y_m}. The covariance matrix P_{y_m} is also known as R in the literature, the *measurement noise*.

3.1 Kalman gain and sensor fusion of normal distributions

Let us rewrite again (8) and (9)

$$\begin{cases} \hat{x}_f &= \frac{\hat{x}_1\sigma_{x_2}^2 + \hat{x}_2\sigma_{x_1}^2}{\sigma_{x_1}^2 + \sigma_{x_2}^2} \\ \sigma_f^2 &= \frac{\sigma_{x_1}^2\sigma_{x_2}^2}{\sigma_{x_1}^2 + \sigma_{x_2}^2} \end{cases}. \tag{21}$$

$$\begin{cases} \hat{x}_f &= \hat{x}_1 + k(\hat{x}_2 - \hat{x}_1) \\ \sigma_f^2 &= \sigma_{x_1}^2 - k\sigma_{x_1}^2 \end{cases}, \tag{22}$$

where

$$k = \frac{\sigma_{x_1}^2}{\sigma_{x_1}^2 + \sigma_{x_2}^2}. \tag{23}$$

These (21) and (22) have been derived for the unidimensional case, i.e., we fused two normal *pdf*'s. Now $\hat{q}$ is a vector, and P is a matrix, instead of $\hat{x}$ and σ_x^2. Let us consider the subscript 1 as our prediction, and 2 as our measurement to extend (22) in matrix form. We first note that $\sigma_{x_1}^2$ would be extended to the matrix form $P_y = HPH^T$ and, similarly, $\sigma_{x_2}^2$ would be extended to the matrix form P_{y_m}. Therefore we can extend k in (23) to the following matrix form

$$\begin{aligned} P_y(P_y + P_{y_m})^{-1} &= HPH^T(P_y + P_{y_m})^{-1} \\ &= HK. \end{aligned} \tag{24}$$

We note that for the unidimensional $q \in \mathbb{R}$ case where we compare an unidimensional observation with an unidimensional measurement, then we have that $H = 1$, hence (23) can be derived from the general *Kalman gain* (24). Now we can extend the second equation in (22) as

$$HP_fH^T = HPH^T - HPH^T(P_y + P_{y_m})^{-1}HPH^T, \tag{25}$$

and noting that we are multiplying by H^T and H at the right and left-hand sides of each term in (25), then we have that

$$\begin{aligned} P_f &= P - PH^T(P_y + P_{y_m})^{-1}HP \\ &= P - KHP. \end{aligned} \tag{26}$$

Thus, we can extend easily (22) to matrix form the fused mean for the state q as

$$\hat{q}_f = \hat{q}_1 + K(\hat{y}_m - \hat{y}). \tag{27}$$

The equations (26) and (27) are typically known as the *correction* step in a Kalman filter.

4 Discrete linear Kalman filter

We can summarize the construction of the discrete linear Kalman filter as follows. Look carefully at the made assumptions! If one of the assumptions is not satisfied, then (rigorously speaking) you cannot continue to the next step.

1. Establish the linear model (11) describing the process that you are interested in. Our states are real numbers at this stage. **We assume that the process is linear**.

2. We replace the representation of our states by *pdf*'s. Instead of having a stacked vector of real numbers, we will have a stacked vector of *pdf*'s. **We assume that the variables follow a normal *pdf*.**

3. The normal *pdf* can be described by its mean and its variance as in (6). We can describe the time evolution of the mean and the variance of a stacked vector of *pdf*'s that goes into the process (11) with (12) and (17) respectively. Note that the uncertainty in your variables (encoded by the covariance matrix P) is growing over time since in (17) we add two positive matrices at every iteration.

4. If P is *big* (by looking at the eigenvalues of interest), then we can confine P by comparing (fusing) an observation (19) with a measurement. Again, both are represented by *pdf*'s. **We assume that the observation can be written as a linear function of q. Therefore, the observation is a (possibly a stacked vector of) normal *pdf* since q is a stacked vector of normal *pdf*'s. We assume that the measurement is a (possibly stacked vector of) normal *pdf* as well**. Then, we can update $\hat{q}$ and P employing (27) and (26) respectively.

4.1 Extended discrete Kalman filter

The assumption on the linearity in the process and observation can be relaxed. Assume that your process is modelled by the non-linear system (10). Then, the non-linear process can be approximated by a linear one for a *short* period of time, e.g., between time intervals Δt. The first order approximation of a non-linear function is done by calculating the corresponding Jacobian matrices

$$F = \frac{\partial f(q,u)}{\partial q} \quad G = \frac{\partial f(q,u)}{\partial u}. \tag{28}$$

If the observation is non-linear, i.e., is modelled by (18), then again, we can use the first order approximation

$$H = \frac{\partial h(q)}{\partial q}. \tag{29}$$

5 Assignments / Examples

Check the Python code to complete the exercises at `https://github.com/noether/kalman`.

5.1 Unidimensional car with a biased accelerometer

Consider a vehicle moving along the horizontal axis x. The car is equipped with an accelerometer and a GPS receptor that can measure the acceleration on the car and the position on the x axis respectively. Both sensors can be characterized by a normal *pdf* where we have the fixed $\sigma_a = 0.01 m/s^2$ and $\sigma_{\text{gps}} = 2m$. The accelerometer can be read every millisecond, and the GPS

every second. We are interested in estimating the position of the vehicle by exploiting both sensors.

From physics we have that

$$x(k+1) = x(k) + v(k)\Delta T + a(k)\frac{\Delta T^2}{2} \tag{30}$$

$$v(k+1) = v(k) + a(k)\Delta T, \tag{31}$$

where $x, v \in \mathbb{R}$ are the position and the velocity of the vehicle respectively, and $\Delta T \in \mathbb{R}$ is the time between two events (when we have a reading of the accelerometer).

We further considered that our accelerometer has not been well calibrated. It has a bias $b \in \mathcal{R}$. We incorporate such a bias to the measured acceleration as follows

$$a_m(k) = a(k) + b(k), \tag{32}$$

where a_m is the reading from the accelerometer. If we further consider that the bias b does not change much over during ΔT, then we can model its time evolution as

$$b(k+1) = b(k), \tag{33}$$

i.e., it remains constant. Now we are ready to write (10) as

$$\Sigma_1 := \begin{cases} x(k+1) & = x(k) + v(k)\Delta T + \big(a_m(k) - b(k)\big)\frac{\Delta T^2}{2} \\ v(k+1) & = v(k) + \big(a_m(k) - b(k)\big)\Delta T \\ b(k+1) & = b(k) \end{cases}, \tag{34}$$

where we can build our state vector $q(k) = \begin{bmatrix} x(k) & v(k) & b(k) \end{bmatrix}^T$.

Check the script `https://github.com/noether/kalman/blob/master/python/assignment_unicar.py` for the (almost complete) implementation of the discrete linear Kalman filter in Section 4 for the system (34). Try to solve or to identify in the script the following questions.

1. Construct F and G as in (11).
2. Given the variance of the accelerometer, construct Q as in (17).
3. Construct H as in (19) for a GPS measurement of x.
4. For the correction step 4, construct/calculate P_{y_m} as in (24).
5. Investigate P whenever you have a GPS measurement available, and decide whether is worth to spend computational power in a correction.
6. Consider there is a radar to measure the velocity v. It follows a normal *pdf*, same frequency as the GPS, and $\sigma_{\text{radar}} = 0.01m/s$. Construct the associated H if we apply the correction step 4 employing GPS and the radar simultaneously.

5.2 Unidimensional car with an accelerometer with a scale factor

Now, we replace (32) by

$$a_m(k) = s(k)a(k), \tag{35}$$

where $s(k) \in \mathbb{R}$ is a constant scale factor close to one. Start a new script based on the previous exercise and try to solve the following questions.

1. Write the new (10). Is it linear? (it is not, why?).
2. Calculate the Jacobians of the new $f(q(k), u(k))$.
3. As before, you can consider GPS and velocity radar for the correction step.

5.3 2D localization (navigation) with range measurements

Check the (incomplete) script `https://github.com/noether/kalman/blob/master/python/assignment_localization_kalman.py`.

We consider a home robot vacuum cleaner that wants to come back to its docking station. The information that we can obtain from the system is the velocity of the robot, e.g., odometry at 1 KHz with a standard deviation of 0.1 m/s for each velocity component. We can also measure the distance of the robot from the docking station, e.g., from a radio chip at 1 Hz with a standard deviation of 0.3 m.

We model the robot as a kinematic point in continuous time

$$\dot{p} = u, \tag{36}$$

where $p, u \in \mathbb{R}^2$ are the position of the robot with respect to the docking station and the control action on the robot respectively. Consequently, the control action u commands the velocity of the robot.

Consider that the state vector $q = p$. Also consider that the robot is commanded with the input signal $u = \begin{bmatrix} 5\sin(t) \\ 10\cos(t) \end{bmatrix}$. To construct the observation function, note that the range measurement is given by $y_m = \sqrt{p^T p} = \sqrt{p_x^2 + p_y^2}$.

1. Construct a **discrete** linear Kalman filter to estimate the position of the robot with respect to the docking station.