卡尔曼滤波

kalman_intro.pdf (unc.edu)

Microsoft Word - Kalman15.doc (yale.edu)

> The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error.

The Discrete Kalman Filter

The Process to be Estimated

$$x_k = Ax_k - 1 + Bu_k - 1 + w_k - 1$$

$$z_k = Hx_k + v_k$$

The random variables and represent the process and measurement noise (respectively). They are assumed to be independent (of each other), white, and with normal probability distributions

$$p(w) \sim N(0, Q)$$

$$p(v) \sim N(0, R)$$

The Computational Origins of the Filter

define a priori and a posteriori estimate errors as

$$e_k^- \equiv x_k - \hat{} x_k^-$$

$$e_k \equiv x_k - \hat{} x_k$$

$$P_k^- = E[(e_k^-)(e_k^-)^T]$$

$$P_k = E[e_k e_k^T]$$

$$\hat{} x_k = \hat{} x_k^- + K(z_k - H \hat{} x_k^-)$$

残差

$$(z_k - H \hat{} x_k^-)$$

The matrix K is chosen to be the gain or blending factor that minimizes the a posteriori error covariance P_k

对迹求导:

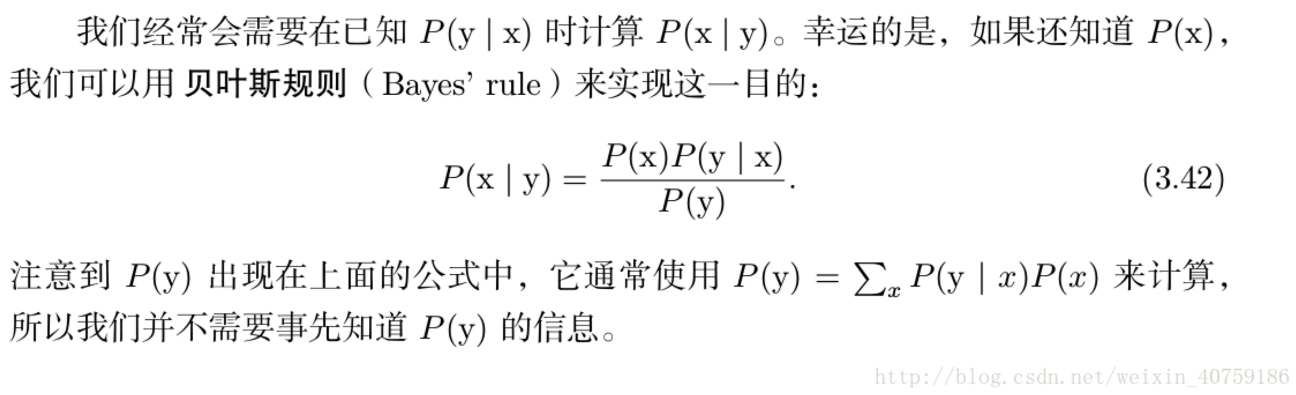
$$K_k = P_k^- H^T (HP_k^- H^T + R)^-1$$

$$lim(R_k \to 0)K_k = H^-1$$

$$lim(P_k^- \to 0)K_k = 0$$

The Probabilistic Origins of the Filter

Bayes’ rule



我们经常会需要在已知 $P(\mathrm{y} \mid \mathrm{x})$ 时计算 $P(\mathrm{x} \mid \mathrm{y})$。幸运的是，如果还知道 $P(\mathrm{x})$，我们可以用 **贝叶斯规则**（Bayes’ rule）来实现这一目的：

$$P(\mathrm{x} \mid \mathrm{y}) = \frac{P(\mathrm{x})P(\mathrm{y} \mid \mathrm{x})}{P(\mathrm{y})}. \tag{3.42}$$

注意到 $P(\mathrm{y})$ 出现在上面的公式中，它通常使用 $P(\mathrm{y}) = \sum_x P(\mathrm{y} \mid x)P(x)$ 来计算，所以我们并不需要事先知道 $P(\mathrm{y})$ 的信息。

后验概率 = 标准相似度*先验概率

卡尔曼滤波满足:

$$E[x_k] = \hat{}x_k$$

$$E[(x_k - \hat{}x_k)(x_k - \hat{}x_k)^T] = P_k.$$

$$p(x_k|z_k) \sim N(E[x_k], E[(x_k - \hat{}x_k)(x_k - \hat{}x_k)^T])$$

$$= N(\hat{}x_k, P_k)$$

The Discrete Kalman Filter Algorithm

the equations for the Kalman filter fall into two groups: time update equations and measurement update equations.

时间更新方程 预测

测量更新方程 校正

The specific equations for the time and measurement updates are presented below in table 4.1 and table 4.2.

Table 4.1: Discrete Kalman filter time update equations.

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \tag{4.9}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{4.10}$$

Again notice how the time update equations in table 4.1 project the state and covariance estimates forward from time step $k-1$ to step k. A and B are from equation (4.1), while Q is from equation (4.3). Initial conditions for the filter are discussed in the earlier references.

Table 4.2: Discrete Kalman filter measurement update equations.

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{4.11}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{4.12}$$

$$P_k = (I - K_k H)P_k^- \tag{4.13}$$

Filter Parameters and Tuning

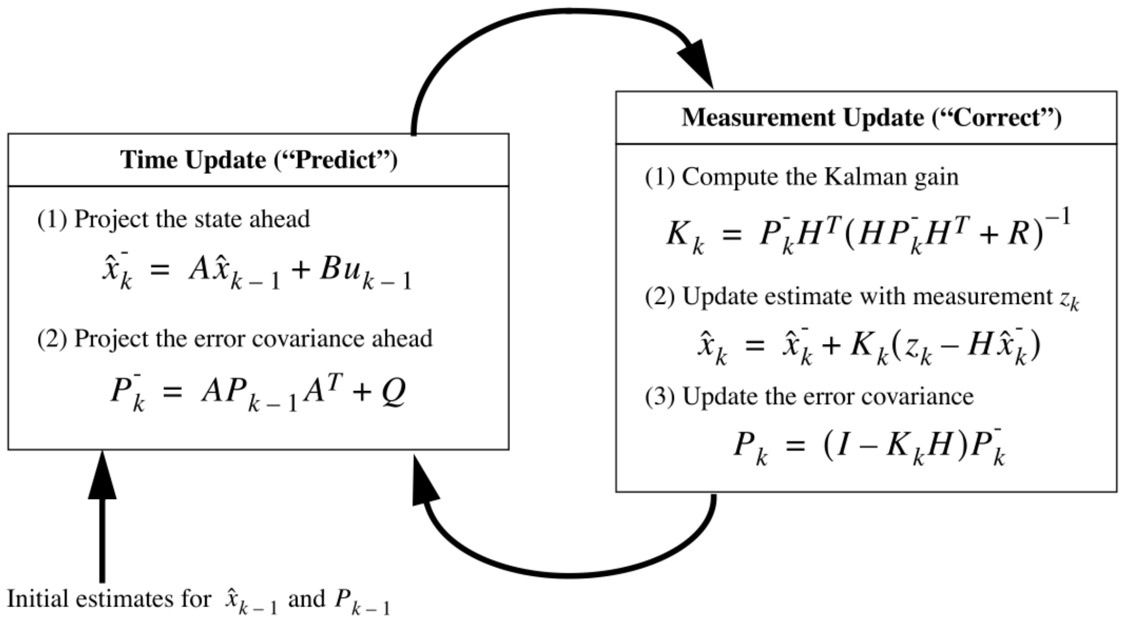

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
(2) Project the error covariance ahead
$P_k^- = AP_{k-1}A^T + Q$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$
(3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Figure 1-2. A complete picture of the operation of the Kalman filter, combining the high-level diagram of Figure 1-1 with the equations from Table 1-1 and Table 1-2.

The Extended Kalman Filter (EKF)

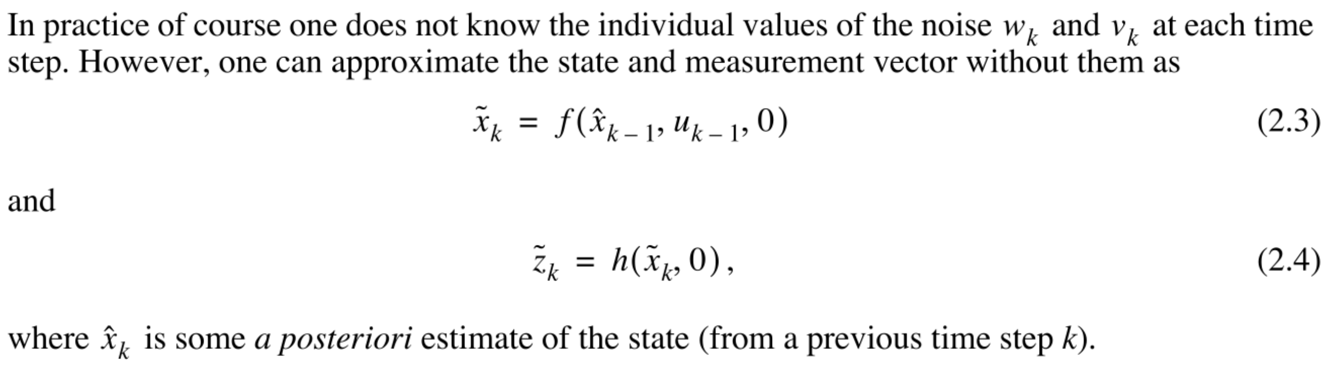
线性化

The Process to be Estimated

$$x_k = f(x_k - 1, u_k - 1, w_k - 1)$$

$$z_k = h(x_k, v_k)$$

It includes as parameters any driving function u_k-1, and the zero-mean process noise w_k. The non-linear function h in the measurement equation relates the state x_k to the measurement z_k
.



In practice of course one does not know the individual values of the noise w_k and v_k at each time step. However, one can approximate the state and measurement vector without them as

$$\tilde{x}_k = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{2.3}$$

and

$$\tilde{z}_k = h(\tilde{x}_k, 0), \tag{2.4}$$

where $\hat{x}_k$ is some *a posteriori* estimate of the state (from a previous time step k).

It is important to note that a fundamental flaw of the EKF is that the distributions (or densities in the continuous case) of the various random variables are no longer normal after undergoing their respective nonlinear transformations. The EKF is simply an ad hoc state estimator that only approximates the optimality of Bayes' rule by linearization.

The Computational Origins of the Filter

$$x_k \approx \tilde{x}_k + A(x_{k-1} - \hat{x}_{k-1}) + Ww_{k-1}, \tag{2.5}$$

$$z_k \approx \tilde{z}_k + H(x_k - \tilde{x}_k) + Vv_k. \tag{2.6}$$

where

- x_k and z_k are the actual state and measurement vectors,
- $\tilde{x}_k$ and $\tilde{z}_k$ are the approximate state and measurement vectors from (2.3) and (2.4),
- $\hat{x}_k$ is an *a posteriori* estimate of the state at step *k*,
- the random variables w_k and v_k represent the process and measurement noise as in (1.3) and (1.4).
- A is the Jacobian matrix of partial derivatives of f with respect to *x*, that is

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}_{k-1}, u_{k-1}, 0),$$

- W is the Jacobian matrix of partial derivatives of f with respect to *w*,

$$W_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{x}_{k-1}, u_{k-1}, 0),$$

- H is the Jacobian matrix of partial derivatives of h with respect to *x*,

$$H_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\tilde{x}_k, 0),$$

- V is the Jacobian matrix of partial derivatives of h with respect to *v*,

$$V_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}}(\tilde{x}_k, 0).$$

[illegible]

测量误差：

$$e^{\sim}(x_k) \equiv x_k - x_k^{\sim}$$

测量残差：

$$e^{\sim}(z_k) \equiv z_k - z_k^{\sim}$$

$$\tilde{e}_{x_k} \approx A(x_{k-1} - \hat{x}_{k-1}) + \varepsilon_k, \tag{2.9}$$

$$\tilde{e}_{z_k} \approx H\tilde{e}_{x_k} + \eta_k, \tag{2.10}$$

where ε and η represent new independent random variables having zero mean and covariance matrices

$$WQW^T, VRV^T$$

$$\hat{}x_k = \tilde{}x_k + \hat{}e_k$$

$$p(\tilde{e}_{x_k}) \sim N(0, E[\tilde{e}_{x_k} \tilde{e}_{x_k}^T])$$
$$p(\varepsilon_k) \sim N(0, WQ_kW^T)$$
$$p(\eta_k) \sim N(0, VR_kV^T)$$

$$\hat{}e_k = K_k \tilde{}e_z k$$

$$\begin{aligned} \hat{x}_k &= \tilde{x}_k + K_k \tilde{e}_{z_k} \\ &= \tilde{x}_k + K_k(z_k - \tilde{z}_k) \end{aligned} \tag{2.13}$$

Table 2-1: EKF time update equations.

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0) \tag{2.14}$$

$$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T \tag{2.15}$$

As with the basic discrete Kalman filter, the time update equations in Table 2-1 project the state and covariance estimates from the previous time step $k-1$ to the current time step k. Again f in (2.14) comes from (2.3), A_k and W_k are the process Jacobians at step *k*, and Q_k is the process noise covariance (1.3) at step *k*.

Table 2-2: EKF measurement update equations.

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1} \tag{2.16}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0)) \tag{2.17}$$

$$P_k = (I - K_k H_k) P_k^- \tag{2.18}$$

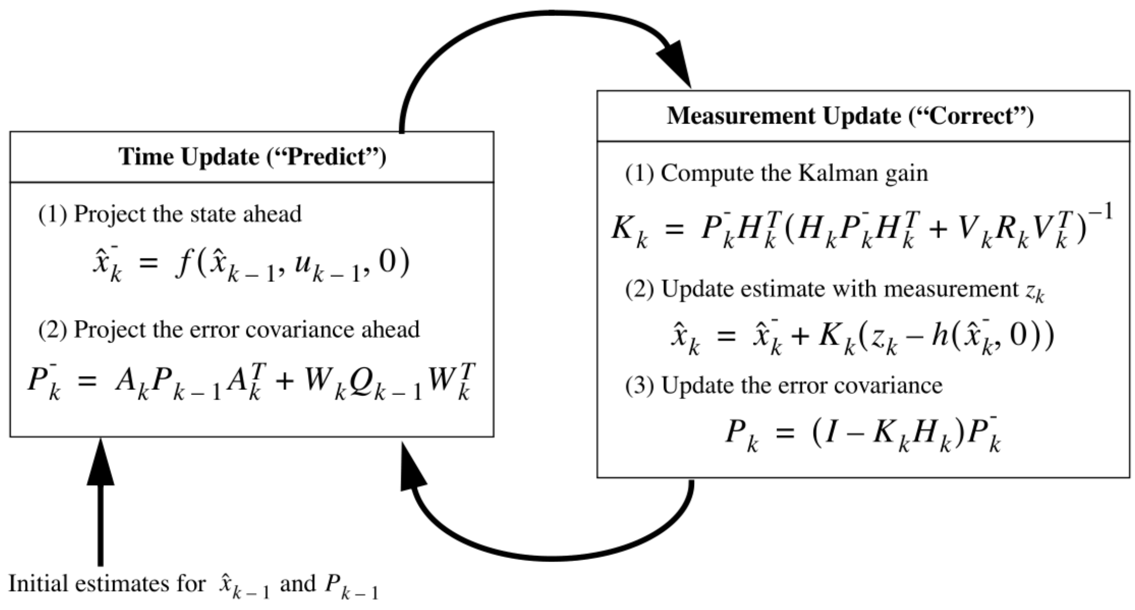
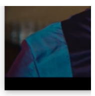
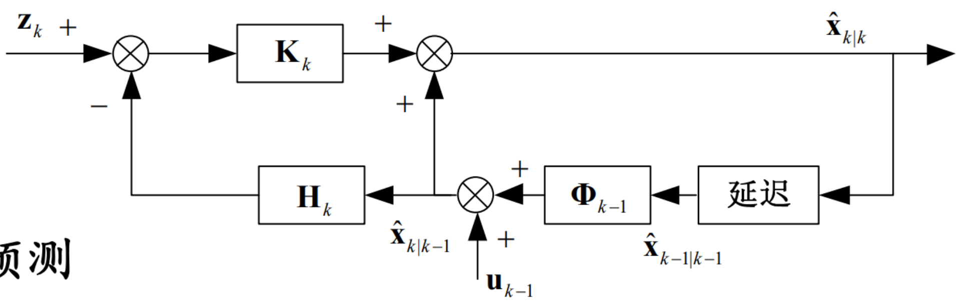

Time Update ("Predict")
(1) Project the state ahead
$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0)$
(2) Project the error covariance ahead
$P_k^- = A_k P_{k-1} A_k^T + W_k Q_{k-1} W_k^T$
Measurement Update ("Correct")
(1) Compute the Kalman gain
$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$
(2) Update estimate with measurement z_k
$\hat{x}_k = \hat{x}_k^- + K_k(z_k - h(\hat{x}_k^-, 0))$
(3) Update the error covariance
$P_k = (I - K_k H_k) P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}

Figure 2-1. A complete picture of the operation of the *extended* Kalman filter, combining the high-level diagram of Figure 1-1 with the equations from Table 2-1 and Table 2-2.



◻ 卡尔曼滤波器算法总结



1. 状态预测

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{\Phi}_{k-1}\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{u}_{k-1}$$

2. 误差协方差预测

$$\mathbf{P}_{k|k-1} = \mathbf{\Phi}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{\Phi}_{k-1}^{\mathrm{T}} + \mathbf{\Gamma}_{k-1}\mathbf{Q}_{k-1}\mathbf{\Gamma}_{k-1}^{\mathrm{T}}$$

3. 卡尔曼滤波器增益

$$\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}\left(\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k\right)^{-1}$$

4. 状态估计校正

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\left(\mathbf{z}_k - \hat{\mathbf{z}}_{k|k-1}\right)$$

其中 $\hat{\mathbf{z}}_{k|k-1} = \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$

5. 误差协方差估计校正

$$\mathbf{P}_{k|k} = \left(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k\right)\mathbf{P}_{k|k-1}$$

RM

在三维空间中进行状态估计，需要位置和速度，x_k有六维。

使用视觉传感器能得到图像的像素点位置，理论上应将像素点作为测量值，实际中对三维点坐标作非线性变换最后用yaw,pitch,distance作为测量量：枪管正对的角度不一定是陀螺仪的0度，计算得到的x,y值方差会跟随枪管正对的陀螺仪角度变化的

P:状态估计协方差,一般设定为单位阵让它自己收敛即可。

R:为测量噪声矩阵,一般只需要设置其对角元素,通常不会将R矩阵的对角线元素设为方差 σ^2,而是设置为 (3σ)^2。

一种做法：敌方装甲板不动，动云台，保存相关的原始坐标值，然后利用Matlab或者Python计算出方差填入该矩阵中。

由于相机的小孔成像模型，如果误差一个像素的话，敌方装甲板在不同距离所计算得到的distance方差是完全不同的。因此，一种显然的想法是根据distance的计算值动态改变distance的方差。目前的做法是测出不同距离的distance方差，然后用一条直线拟合出来，每次都不断计算更新distance的方差。yaw和pitch的方差与距离无关，设置固定值。

Q:如果仅设置Q矩阵的对角阵，默认了一个轴的速度与其位置值无关，这个命题显然是不成立的，速度值和位置值的过程噪声必然存在一些关联。得到同一个轴之间的关联之后，需要调节的参数就只剩下三个了，分别是x,y,z轴的过程噪声。x,y轴应该是等效的(因为存在陀螺仪绕z轴的旋转)，那么x,y轴的过程噪声也应当设置为一致的，即此时需要调节的仅有两个参数。

调参流程：滤波部分和预测部分。

运动模型：

CV：匀速模型（Constant Velocity）

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

状态转移方程：

$$\overrightarrow{x}(t+\Delta t) = \begin{pmatrix} x(t)+\Delta t v_x \\ y(t)+\Delta t v_y \\ v_x \\ v_y \end{pmatrix}$$

另一版本：

$$^{\mathrm{CV}}\mathbf{x} = \begin{pmatrix} x & y & \vartheta & v \end{pmatrix}^{\mathrm{T}}$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \begin{pmatrix} v\cos(\vartheta)T_k \\ v\sin(\vartheta)T_k \\ 0 \\ 0 \end{pmatrix}.$$

对于$x=(x,y,v_x,v_y)^T$，推导运动模型以及状态转移矩阵。

$$\begin{aligned} x_{k+1} &= A_k x_k + Bu_k + \omega_k \\ z_{k+1} &= Hx_{k+1} + \upsilon_{k+1} \end{aligned}$$

$$x_{k+1} = \begin{Bmatrix} x \\ y \\ v_x \\ v_y \end{Bmatrix}_{k+1} = \begin{Bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{Bmatrix} \begin{Bmatrix} x \\ y \\ v_x \\ v_y \end{Bmatrix}_k$$

噪声(根据s=1/2aΔt^2):

$$\omega = \begin{Bmatrix} \frac{1}{2}a_x\Delta t^2 \\ \frac{1}{2}a_y\Delta t^2 \\ a_x\Delta t \\ a_y\Delta t \end{Bmatrix} = \begin{Bmatrix} \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{Bmatrix} \begin{Bmatrix} a_x \\ a_y \end{Bmatrix} = Gu$$

$$u = \begin{Bmatrix} a_x \\ a_y \end{Bmatrix} \quad G = \begin{Bmatrix} \frac{\Delta t^2}{2} & 0 \\ 0 & \frac{\Delta t^2}{2} \\ \Delta t & 0 \\ 0 & \Delta t \end{Bmatrix}$$

$$Q = \mathrm{cov}(\omega) = E(\omega\omega^T) = GE(uu^T)G^T = G\begin{Bmatrix} \sigma_{a_x}^2 & 0 \\ 0 & \sigma_{a_y}^2 \end{Bmatrix} G^T$$

$$Q = \begin{Bmatrix} 0.25\Delta t^4{\sigma_{a_x}}^2 & 0 & 0.5\Delta t^3{\sigma_{a_x}}^2 & 0 \\ 0 & 0.25\Delta t^4{\sigma_{a_y}}^2 & 0 & 0.5\Delta t^3{\sigma_{a_y}}^2 \\ 0.5\Delta t^3{\sigma_{a_x}}^2 & 0 & \Delta t^2{\sigma_{a_x}}^2 & 0 \\ 0 & 0.5\Delta t^3{\sigma_{a_y}}^2 & 0 & \Delta t^2{\sigma_{a_y}}^2 \end{Bmatrix}$$

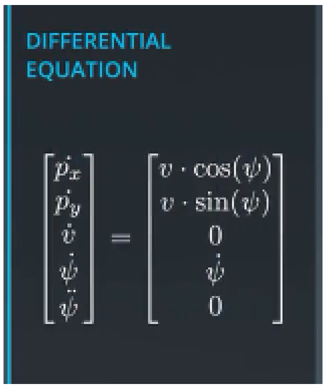
CA:匀加速模型（Constant Acceleration)

$${}^{CA}\mathbf{x} = \begin{pmatrix} x & y & \vartheta & v & a \end{pmatrix}^T.$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \begin{pmatrix} \left(v_k T_k + \frac{a}{2}T_k^2\right)\cos(\vartheta) \\ \left(v_k T_k + \frac{a}{2}T_k^2\right)\sin(\vartheta) \\ 0 \\ aT_k \\ 0 \end{pmatrix}.$$

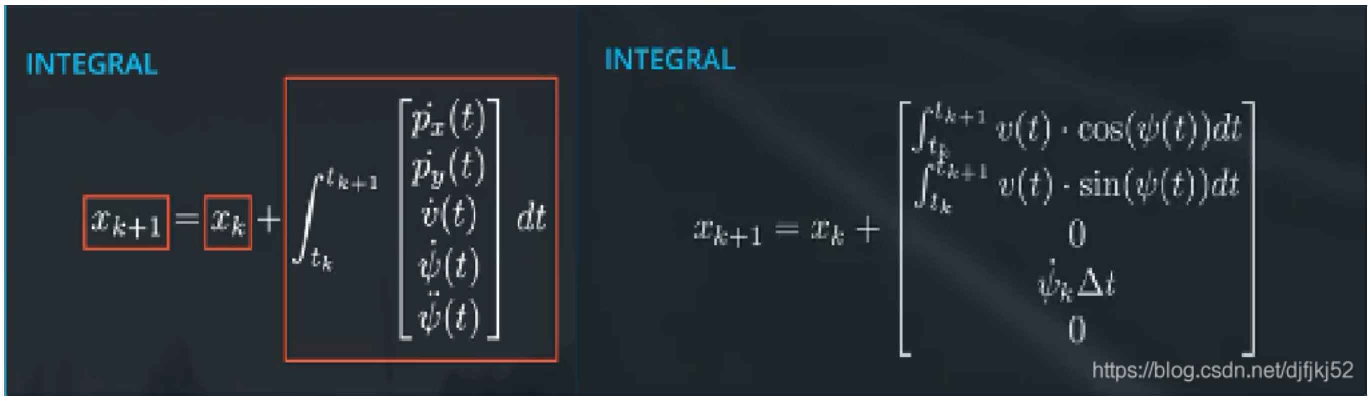
CTRV: 恒定转弯率和速度幅度模型（Constant Turn Rate and Velocity)

$$x = \begin{bmatrix} p_x \\ p_y \\ v \\ \psi \\ \dot{\psi} \end{bmatrix}$$



DIFFERENTIAL EQUATION

$$\begin{bmatrix} \dot{p_x} \\ \dot{p_y} \\ \dot{v} \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} v \cdot \cos(\psi) \\ v \cdot \sin(\psi) \\ 0 \\ \dot{\psi} \\ 0 \end{bmatrix}$$



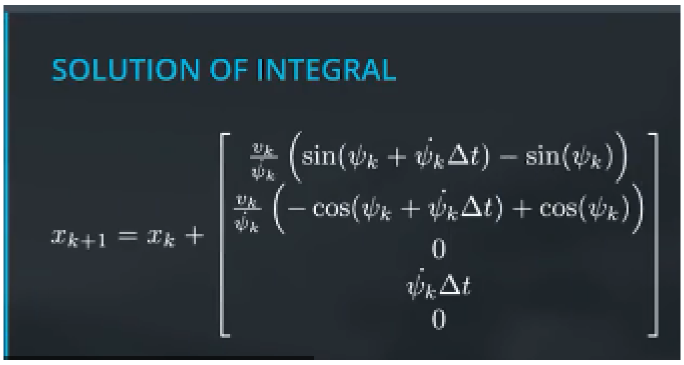
INTEGRAL

$$\boxed{x_{k+1}} = \boxed{x_k} + \boxed{\int_{t_k}^{t_{k+1}} \begin{bmatrix} \dot{p_x}(t) \\ \dot{p_y}(t) \\ \dot{v}(t) \\ \dot{\psi}(t) \\ \ddot{\psi}(t) \end{bmatrix} dt}$$

INTEGRAL

$$x_{k+1} = x_k + \begin{bmatrix} \int_{t_k}^{t_{k+1}} v(t) \cdot \cos(\psi(t))dt \\ \int_{t_k}^{t_{k+1}} v(t) \cdot \sin(\psi(t))dt \\ 0 \\ \dot{\psi}_k \Delta t \\ 0 \end{bmatrix}$$

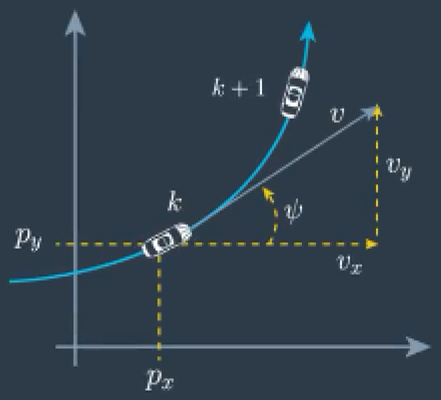
积分：



SOLUTION OF INTEGRAL

$$x_{k+1} = x_k + \begin{bmatrix} \frac{v_k}{\dot{\psi}_k}\left(\sin(\psi_k + \dot{\psi}_k \Delta t) - \sin(\psi_k)\right) \\ \frac{v_k}{\dot{\psi}_k}\left(-\cos(\psi_k + \dot{\psi}_k \Delta t) + \cos(\psi_k)\right) \\ 0 \\ \dot{\psi}_k \Delta t \\ 0 \end{bmatrix}$$

噪声：



PROCESS MODEL

$$x_{k+1} = f(x_k, \nu_k)$$

NOISE VECTOR

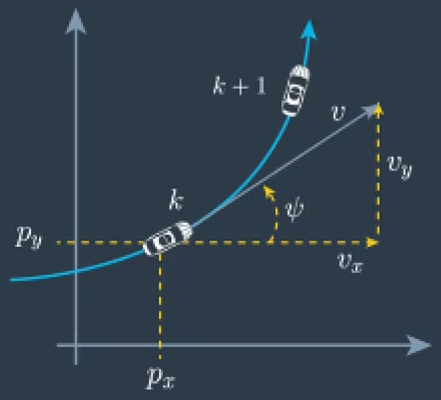
$$\nu_k = \begin{bmatrix} \nu_{a,k} \\ \nu_{\ddot{\psi},k} \end{bmatrix}$$

LONGITUDINAL ACCELERATION NOISE:
STATISTICAL PROPERTIES

$$\nu_{a,k} \sim \mathcal{N}(0,\, \sigma_a^2)$$

YAW ACCELERATION NOISE:
STATISTICAL PROPERTIES

$$\nu_{\ddot{\psi},k} \sim \mathcal{N}(0,\, \sigma_{\ddot{\psi}}^2)$$



STATE VECTOR

$$x = \begin{bmatrix} p_x \\ p_y \\ v \\ \psi \\ \dot{\psi} \end{bmatrix}$$

PROCESS NOISE VECTOR

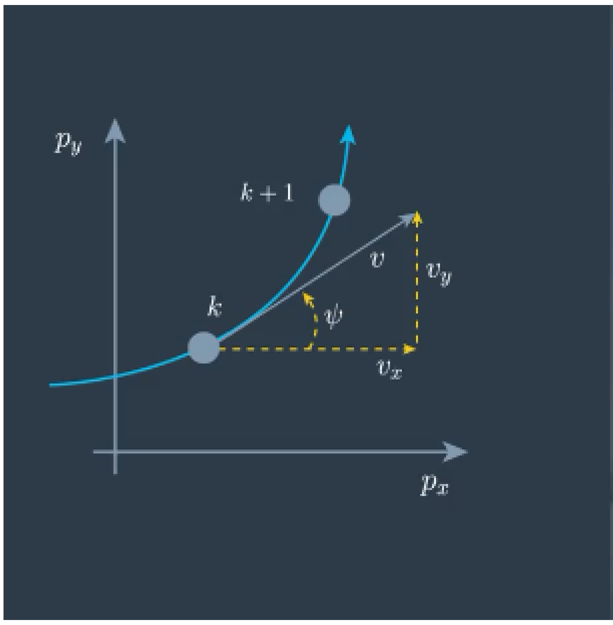
$$\nu_k = \begin{bmatrix} \nu_{a,k} \\ \nu_{\ddot{\psi},k} \end{bmatrix}$$

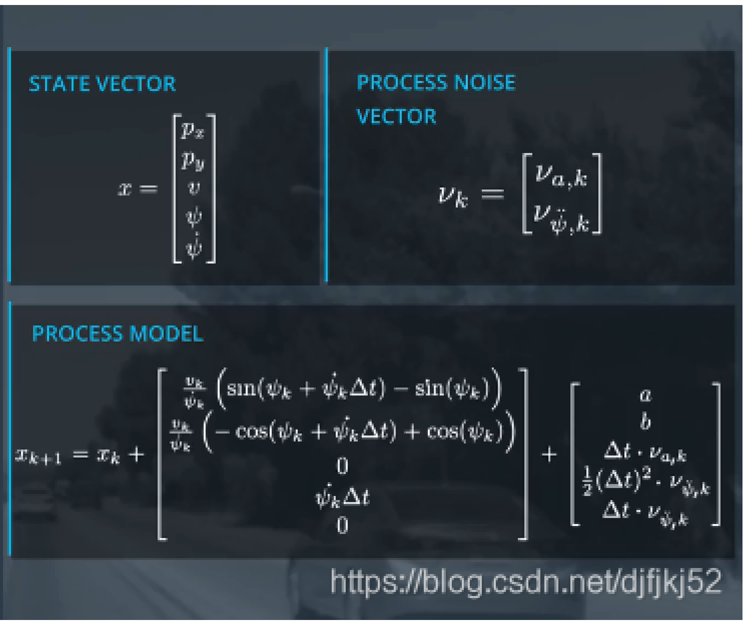
PROCESS MODEL

$$x_{k+1} = x_k + \begin{bmatrix} \frac{v_k}{\dot{\psi}_k}\Big(\sin(\psi_k + \dot{\psi}_k \Delta t) - \sin(\psi_k)\Big) \\ \frac{v_k}{\dot{\psi}_k}\Big(-\cos(\psi_k + \dot{\psi}_k \Delta t) + \cos(\psi_k)\Big) \\ 0 \\ \dot{\psi}_k \Delta t \\ 0 \end{bmatrix} + \begin{bmatrix} a \\ b \\ c \\ d \\ e \end{bmatrix}$$

INFLUENCE YAW ACCELERATION ON YAW RATE

$$e = \Delta t \cdot \nu_{\ddot{\psi},k}$$





STATE VECTOR

$$x = \begin{bmatrix} p_x \\ p_y \\ v \\ \psi \\ \dot{\psi} \end{bmatrix}$$

PROCESS NOISE VECTOR

$$\nu_k = \begin{bmatrix} \nu_{a,k} \\ \nu_{\ddot{\psi},k} \end{bmatrix}$$

PROCESS MODEL

$$x_{k+1} = x_k + \begin{bmatrix} \frac{v_k}{\dot{\psi}_k}\left(\sin(\psi_k + \dot{\psi}_k \Delta t) - \sin(\psi_k)\right) \\ \frac{v_k}{\dot{\psi}_k}\left(-\cos(\psi_k + \dot{\psi}_k \Delta t) + \cos(\psi_k)\right) \\ 0 \\ \dot{\psi}_k \Delta t \\ 0 \end{bmatrix} + \begin{bmatrix} a \\ b \\ \Delta t \cdot \nu_{a,k} \\ \frac{1}{2}(\Delta t)^2 \cdot \nu_{\ddot{\psi},k} \\ \Delta t \cdot \nu_{\ddot{\psi},k} \end{bmatrix}$$

CTRA:**恒定转动率和加速度**（Constant Turn Rate and Acceleration)

$$^{\mathrm{CTRA}}\mathbf{x} = \begin{pmatrix} x & y & \vartheta & v & a & \omega \end{pmatrix}^{\mathrm{T}}.$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \begin{pmatrix} g_x(\mathbf{x}_k, T_k) \\ g_y(\mathbf{x}_k, T_k) \\ \omega T_k \\ a T_k \\ 0 \\ 0 \end{pmatrix},$$

$$\begin{aligned} g_x = {} & \frac{a[\cos(\vartheta_k + \omega T_k) - \cos(\vartheta_k)]}{\omega^2} \\ & + \frac{(v_k + a T_k)\sin(\vartheta_k + \omega T_k) - v_k \sin(\vartheta_k)}{\omega}, \\ g_y = {} & \frac{a[\sin(\vartheta_k + \omega T_k) - \sin(\vartheta_k)]}{\omega^2} \\ & - \frac{(v_k + a T_k)\cos(\vartheta_k + \omega T_k) - v_k \cos(\vartheta_k)}{\omega}. \end{aligned}$$

恒定转向角和速度（CSAV)

CCA:**恒定曲率和加速度**(Constant Curvature and Acceleration)