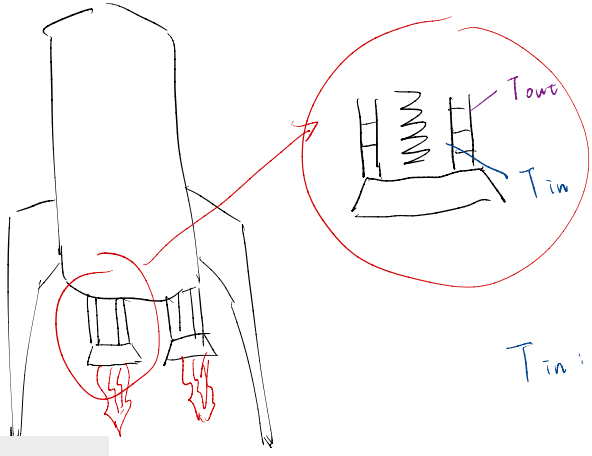
• Why we use Kalman Filters. ?

卡尔曼滤波器

例子1：



Tout: Aavailable

Tin: Not available for measurement

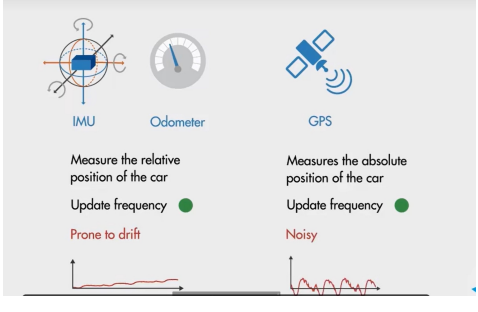
• The variables of interest can only be measured indirectly.

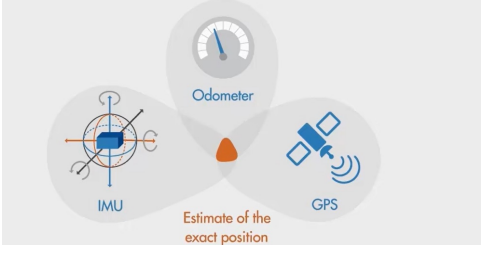
Kalman Filter 利用间接测量值，计算内部温度的最优估算值.

Optimal estimation algorithm.

例子2：

• Measurements are available from various sensors but might be subject to noise.

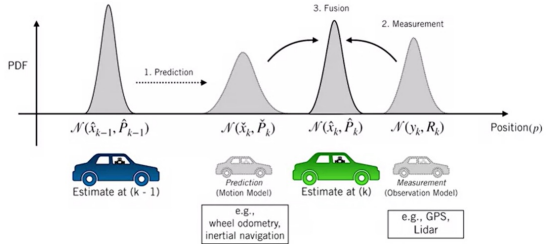




有 Noise.

Linear Kalman Filter

The Kalman Filter I Prediction and Correction



The Kalman Filter I Linear Dynamical System

- The Kalman Filter requires the following motion and measurement models:

Motion model: $\mathbf{x}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{u}_{k-1} + \mathbf{w}_{k-1}$ (input, noise)

Measurement model: $\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{v}_k$ (noise)

- With the following noise properties:

$\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ Measurement Noise

$\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_k)$ Process or Motion Noise

The Kalman Filter I Recursive Least Squares + Process Model

- The Kalman filter is a recursive least squares estimator that also includes a motion model

1 Prediction

$$\check{\mathbf{x}}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{u}_{k-1}$$
$$\check{\mathbf{P}}_k = \mathbf{F}_{k-1}\hat{\mathbf{P}}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}$$

2b Correction

$$\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{y}_k - \mathbf{H}_k\check{\mathbf{x}}_k)$$
$$\hat{\mathbf{P}}_k = (\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)\check{\mathbf{P}}_k$$

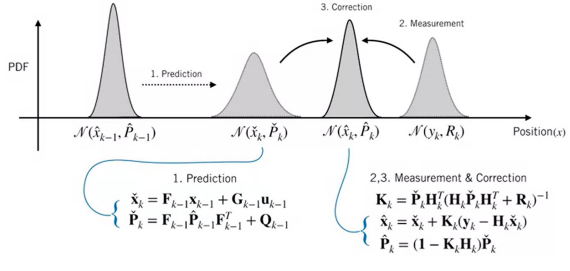
$(\mathbf{y}_k - \mathbf{H}_k\check{\mathbf{x}}_k)$ is often called the 'innovation'

2a Optimal Gain

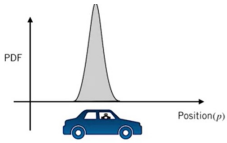
$$\mathbf{K}_k = \check{\mathbf{P}}_k\mathbf{H}_k^T(\mathbf{H}_k\check{\mathbf{P}}_k\mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

$\check{\mathbf{x}}_k$ **Prediction** *(given motion model) at time k*

$\hat{\mathbf{x}}_k$ **Corrected prediction** (given measurement) at time k



Ex:



太抽象了!!!

下一面将详细讲述例子1来源以下网站

https://www.bzarg.com/p/how-a-kalman-filter-works-in-pictures/

我们已知 位置 和 其一阶导数速度

$$X = \begin{bmatrix} p \\ \frac{dp}{dt} = \dot{p} \end{bmatrix}$$

我们输入量则为 加速度

$$u = a = \frac{d^2p}{dt^2}$$

- Motion / Process Model

$$X_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} X_{k-1} + \begin{bmatrix} 0 \\ \Delta t \end{bmatrix} u_{k-1} + w_{k-1}$$

- Position Observation

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} X_k + v_k$$

- Noise Densities

$$v_k \sim \mathcal{N}(0,\ 0.05) \qquad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, (0.1)\mathbf{1}_{2\times 2})$$

- Data

$$\hat{\mathbf{x}}_0 \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 5 \end{bmatrix}, \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

$$\Delta t = 0.5\text{s}$$

$$u_0 = -2\ [m/s^2] \qquad y_1 = 2.2\ [m]$$

Prediction

$$\check{\mathbf{x}}_k = \mathbf{F}_{k-1}\mathbf{x}_{k-1} + \mathbf{G}_{k-1}\mathbf{u}_{k-1}$$

$$\begin{bmatrix} \check{p}_1 \\ \check{p}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 5 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix} (-2) = \begin{bmatrix} 2.5 \\ 4 \end{bmatrix}$$

$$\check{\mathbf{P}}_k = \mathbf{F}_{k-1}\hat{\mathbf{P}}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{Q}_{k-1}$$

$$\check{\mathbf{P}}_1 = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix}^T + \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} = \begin{bmatrix} 0.36 & 0.5 \\ 0.5 & 1.1 \end{bmatrix}$$

Correction

$$\mathbf{K}_1 = \hat{\mathbf{P}}_1\mathbf{H}_1^T(\mathbf{H}_1\check{\mathbf{P}}_1\mathbf{H}_1^T + \mathbf{R}_1)^{-1}$$

$$= \begin{bmatrix} 0.36 & 0.5 \\ 0.5 & 1.1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \left(\begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 0.36 & 0.5 \\ 0.5 & 1.1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + 0.05 \right)^{-1}$$

$$= \begin{bmatrix} 0.88 \\ 1.22 \end{bmatrix}$$

$$\hat{\mathbf{x}}_1 = \check{\mathbf{x}}_1 + \mathbf{K}_1(\mathbf{y}_1 - \mathbf{H}_1\check{\mathbf{x}}_1)$$

$$\begin{bmatrix} \hat{p}_1 \\ \hat{p}_1 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 4 \end{bmatrix} + \begin{bmatrix} 0.88 \\ 1.22 \end{bmatrix} (2.2 - \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} 2.5 \\ 4 \end{bmatrix}) = \begin{bmatrix} 2.24 \\ 3.63 \end{bmatrix}$$

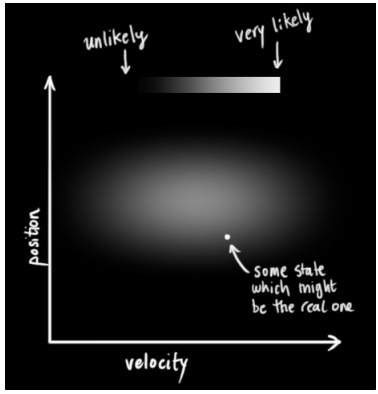
Bonus!

$$\hat{\mathbf{P}}_1 = (\mathbf{1} - \mathbf{K}_1\mathbf{H}_1)\check{\mathbf{P}}_1$$

$$= \begin{bmatrix} 0.04 & 0.06 \\ 0.06 & 0.49 \end{bmatrix}$$

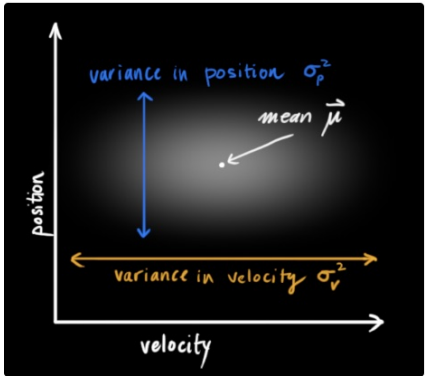
详细例子：

$\vec{x} = \begin{bmatrix} p \\ v \end{bmatrix}$

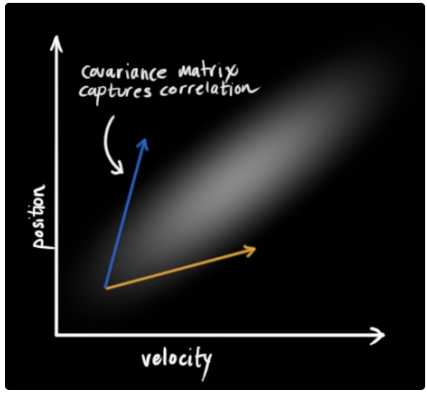


我们不知道相前实际 P 和 V 是多少

Kalman Filter 假设两个变量（P 和 V）都是随机且高斯分布（μ, σ^2）



此图中 P 和 V 不相关，这意味着 V 的状态无法告诉 P 的状态



此图中 P 和 V correlate（covariance）这种关系用协方差矩阵来捕获。

Σ_{ij}

（ith state variable and jth ~ correlate.）

▷ Describing problem with matrice.

we see two pieces of information at k time

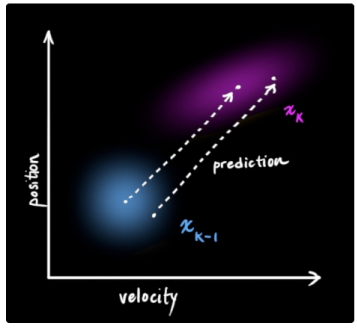
we call best estimate $\hat{x}_k$ (the mean, elsewhere named μ)

and its covariance matrix P_k

$$\hat{x}_k = \begin{bmatrix} \text{position} \\ \text{velocity} \end{bmatrix} \qquad P_k = \begin{bmatrix} \Sigma_{pp} & \Sigma_{pv} \\ \Sigma_{vp} & \Sigma_{vv} \end{bmatrix}$$

当前状态 (k-1)

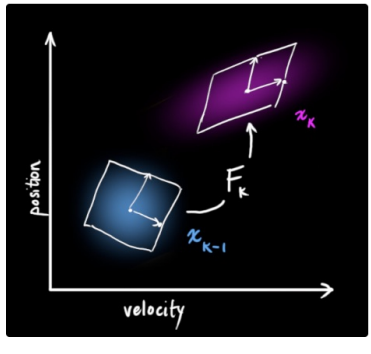
预测状态 (k)



→ 高斯模糊

用矩阵 F_k

表示预测步骤



现在我们有一个预测矩阵，但我们还不知道如何更新 covariance

↑

$$P_k = P_{k-1} + \Delta t \cdot v_{k-1}$$

$$v_k = v_{k-1}$$

$$\Rightarrow \hat{x}_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \hat{x}_{k-1}$$

$$= F_k \cdot \hat{x}_{k-1}$$

$$Cov(x) = \Sigma$$

$$Cov(Ax) = A \Sigma A^T$$

$$\Rightarrow \quad \hat{X}_k = F_k \hat{X}_{k-1}$$

$$P_k = F_k P_{k-1} F_k^T$$

△ External influence.
外部世界会影响系统 (计算到)
比如：加速度 a.

$$\begin{cases} P_k = P_{k-1} + \Delta t\, v_{k-1} + \frac{1}{2} a \Delta t^2 \\ v_k = v_{k-1} + a \Delta t \end{cases}$$

In matrix form:

$$\hat{X}_k = F_k \hat{X}_{k-1} + \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} a$$

$$= F_k \hat{X}_{k-1} + B_k \vec{u}_k$$

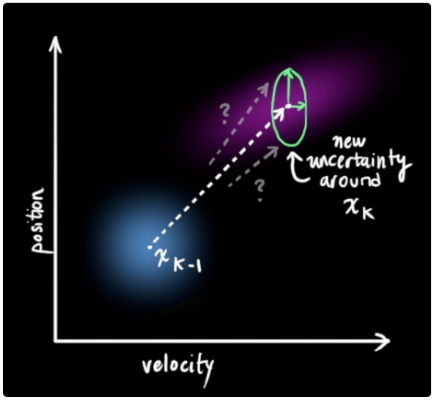
B_k : control matrix

$\vec{u}_k$: control vector

△ External uncertainty.

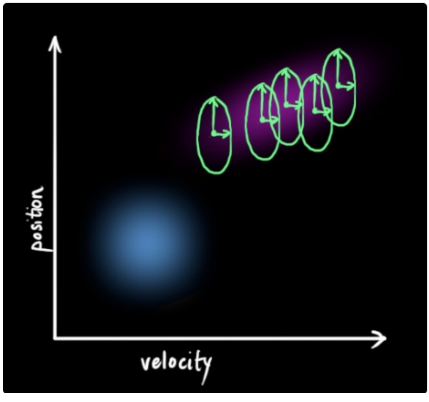
比如：无人机会受到风的影响.

轮式机器人，轮子会打滑.



each point in $\hat{x}_{k-1}$ move to somewhere inside 高斯点 with covariance Q_K

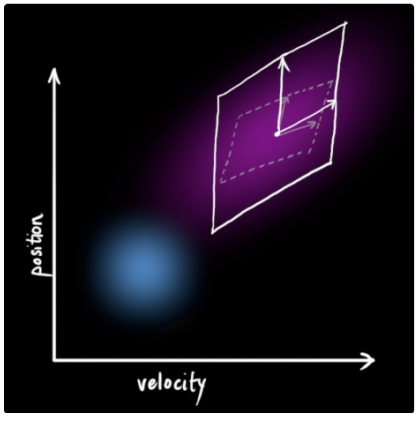
另一种说法是我们将未跟踪的 influence 视为 noise with covariance Q_k



different covariance

(but the same mean)

We get the expanded covariance by simple adding Q_k.



$$\hat{x}_K = F_k \hat{x}_{k-1} + B_k \vec{u}_k$$

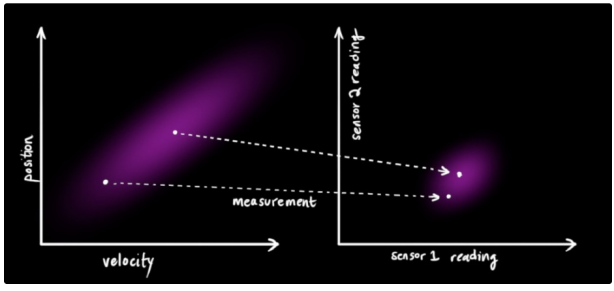
$$P_K = F_k P_{k-1} F_k^T + Q_k$$

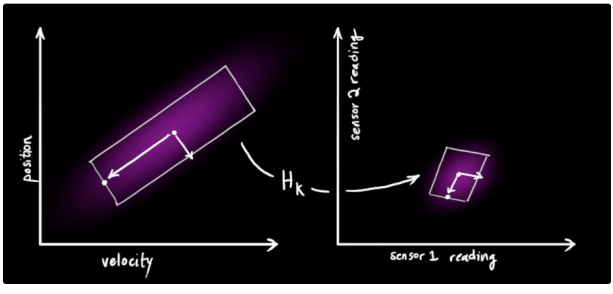
换句话说，新的最佳估计是根据先前的最佳估计做出的预测，加上对已知外部影响的修正。

新的不确定性是根据旧的不确定性预测的，还有一些来自环境的额外不确定性。

Refining the estimate with measurements.

我们可能有几个传感器可以为我们提供有关系统状态的信息。目前，他们测量什么并不重要；重要的是。也许一个读取位置，另一个读取速度。每个传感器都会**间接**告诉我们有关状态的信息，换句话说，传感器在一个状态上运行并产生一组**读数**。



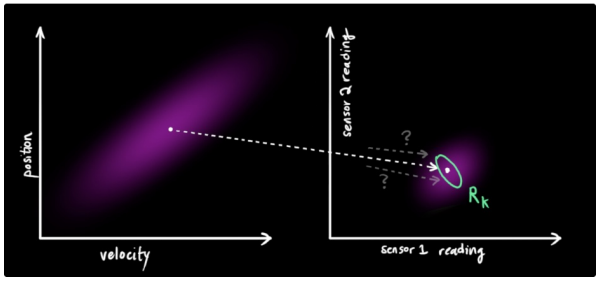


读数的单位和比例可能与我们跟踪的状态和比例不同

∴ 我们使用矩阵对传感器进行建模 H_k.

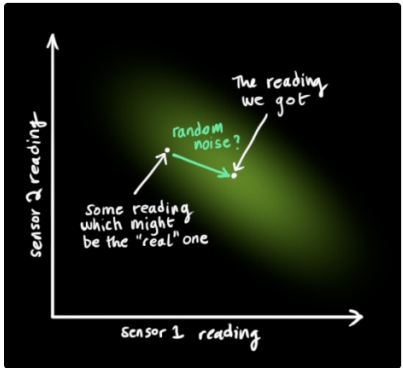
$$\vec{\mu}_{expected} = H_k \hat{x}_k$$

$$\Sigma_{expected} = H_k P_k H_k^T$$



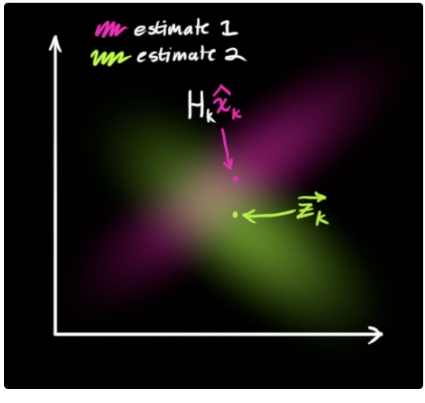
由于存在不确定性

一些 state 比其它 state 更有可能产生我们所看到的读数



将这种不确定性（传感器噪声）的 covariance 标为 R_k.

该平均告知的 mean 等于 the reading we observed, which called $\vec{z}_k$.

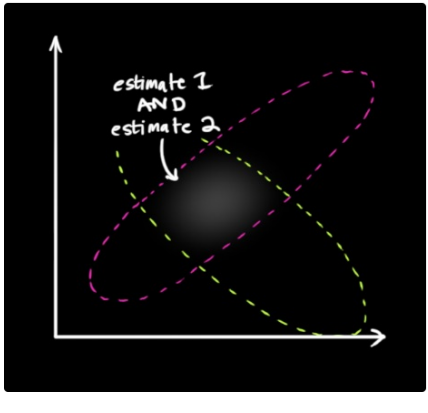


现在我们有两个高斯的点

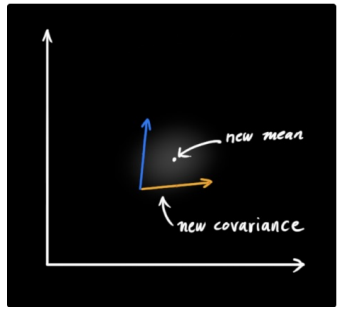
一个是转换后的预测平均值

一个是围绕我们得到的实际估感器读数

预测　　传感器读数



两个概率都为真，相乘，

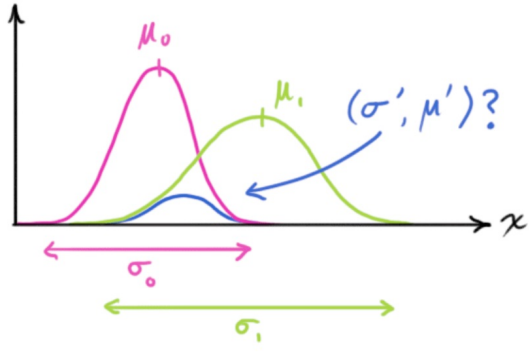


得到重叠（新的高斯点）

Combining Gaussians

Let's find that formula. It's easiest to look at this first in **one dimension**. A 1D Gaussian bell curve with variance σ^2 and mean μ is defined as:

$$\mathcal{N}(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{9}$$



两个高斯曲线相乘

得到蓝色。

$$\mathcal{N}(x, \mu_0, \sigma_0) \cdot \mathcal{N}(x, \mu_1, \sigma_1) \stackrel{?}{=} \mathcal{N}(x, \mu', \sigma') \tag{10}$$

$$\mu' = \mu_0 + \frac{\sigma_0^2 (\mu_1 - \mu_0)}{\sigma_0^2 + \sigma_1^2}$$

$$\sigma'^2 = \sigma_0^2 - \frac{\sigma_0^4}{\sigma_0^2 + \sigma_1^2}$$

Let $k = \frac{\sigma_0^2}{\sigma_0^2 + \sigma_1^2}$

$$\mu' = \mu_0 + k(\mu_1 - \mu_0)$$

$$\sigma'^2 = \sigma_0^2 - k\sigma_0^2$$

矩阵版本：

$$k = \Sigma_0 (\Sigma_0 + \Sigma_1)^{-1}$$

$$\vec{\mu}' = \vec{\mu}_0 + k(\vec{\mu}_1 - \vec{\mu}_0)$$

$$\Sigma' = \Sigma_0 - k\Sigma_0$$

（k是 Kalman gain）

Putting together

two distribution :

- The predicted measurement with $(\mu_0, \Sigma_0) = (H_k \hat{x}_k, H_k P_k H_k^T)$
- observed measurement with $(\mu_1, \Sigma_1) = (\vec{z}_k, R_k)$

$$H_k \hat{x}_k' = H_k \hat{x}_k + k(\vec{z}_k - H_k \hat{x}_k)$$

$$H_k P_k' H_k^T = H_k P_k H_k^T - k H_k P_k H_k^T$$

$$\because K = \Sigma_0 (\Sigma_0 + \Sigma_1)^{-1}$$

$$\therefore K = H_k P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$$

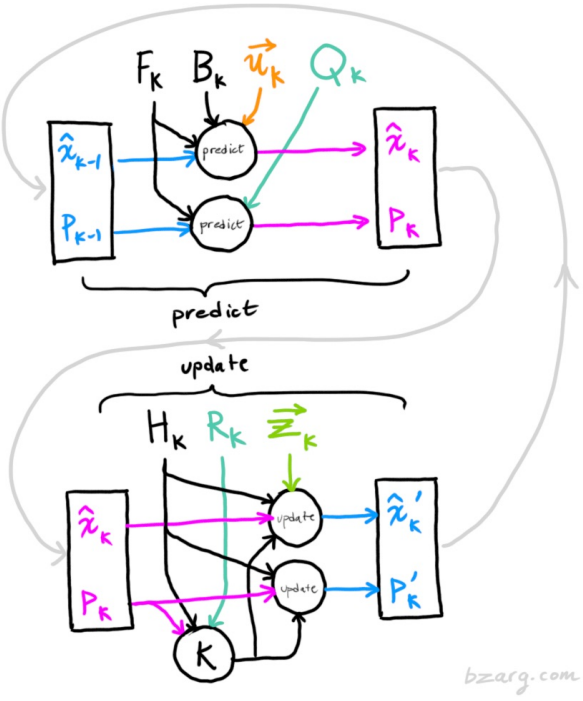
得 $$\hat{x}_k' = \hat{x}_k + k'(\vec{z}_k - H_k \hat{x}_k)$$

$$\hat{P}_k' = P_k - k' H_k P_k$$

$$\left\{ k' = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1} \right\}$$

$\hat{x}_k'$ is new best estimate

Kalman Filter Information Flow



我们需要实现的是以上公式中 3 个

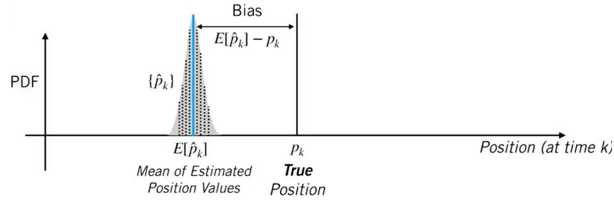
① $$\begin{cases} \hat{x}_k = F_k \hat{x}_{k-1} + B_k \vec{u}_k \\ \hat{P}_k = F_k P_{k-1} F_k^T + Q_k \end{cases}$$

② $$\begin{cases} \hat{x}'_k = \hat{x}_k + k'(\vec{z}_k - H_k \hat{x}_k) \\ P'_k = P_k - k' H_k P_k \end{cases}$$

③ $$k' = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$$

- Kalman Filter and The Bias BLUES

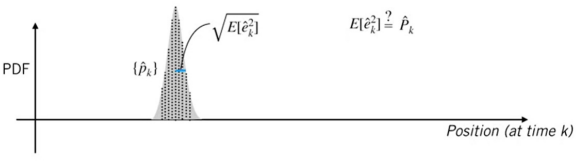
Bias in State Estimation



This filter is an unbiased if **for all *k***,

$$E[\hat{e}_k] = E[\hat{p}_k - p_k] = E[\hat{p}_k] - p_k = 0$$

Consistency in State Estimation



This filter is consistent if **for all *k***,

$$E[\hat{e}_k^2] = E[(\hat{p}_k - p_k)^2] = \hat{P}_k$$

- Consider the *error dynamics*:

$$\check{\mathbf{e}}_k = \check{\mathbf{x}}_k - \mathbf{x}_k \qquad \hat{\mathbf{e}}_k = \hat{\mathbf{x}}_k - \mathbf{x}_k$$

Predicted state error *Corrected estimate error*

- Using the Kalman Filter equations, we can derive:

$$\check{\mathbf{e}}_k = \mathbf{F}_{k-1}\check{\mathbf{e}}_{k-1} - \mathbf{w}_k$$
$$\hat{\mathbf{e}}_k = (\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)\check{\mathbf{e}}_k + \mathbf{K}_k\mathbf{v}_k$$

- For the Kalman filter, for all *k*,

$$E[\check{\mathbf{e}}_k] = E[\mathbf{F}_{k-1}\check{\mathbf{e}}_{k-1} - \mathbf{w}_k] = \mathbf{F}_{k-1}E[\check{\mathbf{e}}_{k-1}] - E[\mathbf{w}_k] = \mathbf{0}$$

$$E[\hat{\mathbf{e}}_k] = E[(\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)\check{\mathbf{e}}_k + \mathbf{K}_k\mathbf{v}_k] = (\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)E[\check{\mathbf{e}}_k] + \mathbf{K}_kE[\mathbf{v}_k] = \mathbf{0}$$

***Unbiased** predictions!*

So long as $E[\hat{\mathbf{e}}_0] = \mathbf{0}$ $E[\mathbf{v}] = \mathbf{0}$ $E[\mathbf{w}] = \mathbf{0}$

+ white, uncorrelated noise

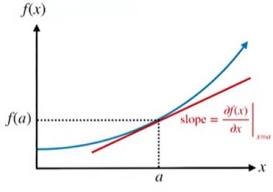
Note: this does *not* mean that the error on a *given* trial will be zero, but that, with enough trials, our expected error is zero!

(Nonlinear) Extended Kalman Filter.

"EKF"

要点：对非线性系统进行线性化处理

Choose an operating point a and approximate the nonlinear function by a tangent line at that point



Mathematically, we compute this linear approximation using a first-order Taylor expansion:

泰勒展开.

$$f(x) \approx f(a) + \underbrace{\left.\frac{\partial f(x)}{\partial x}\right|_{x=a}(x-a)}_{\text{First-order terms}} + \underbrace{\frac{1}{2!}\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x=a}(x-a)^2 + \frac{1}{3!}\left.\frac{\partial^2 f(x)}{\partial x^2}\right|_{x=a}(x-a)^3 + \dots}_{\text{Higher-order terms}}$$

只看第一项

For the EKF, we choose the operating point to be our most recent state estimate, our known input, and zero noise:

Linearized motion model

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \approx \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) + \underbrace{\left.\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{x}_{k-1}}\right|_{\hat{\mathbf{x}}_{k-1},\mathbf{u}_{k-1},\mathbf{0}}}_{\mathbf{F}_{k-1}}(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}) + \underbrace{\left.\frac{\partial \mathbf{f}_{k-1}}{\partial \mathbf{w}_{k-1}}\right|_{\hat{\mathbf{x}}_{k-1},\mathbf{u}_{k-1},\mathbf{0}}}_{\mathbf{L}_{k-1}}\mathbf{w}_{k-1}$$

Linearized measurement model

$$\mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k, \mathbf{v}_k) \approx \mathbf{h}_k(\check{\mathbf{x}}_k, \mathbf{0}) + \underbrace{\left.\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}\right|_{\check{\mathbf{x}}_k,\mathbf{0}}}_{\mathbf{H}_k}(\mathbf{x}_k - \check{\mathbf{x}}_k) + \underbrace{\left.\frac{\partial \mathbf{h}_k}{\partial \mathbf{v}_k}\right|_{\check{\mathbf{x}}_k,\mathbf{0}}}_{\mathbf{M}_k}\mathbf{v}_k$$

We now have a linear system in state-space! The matrices $\mathbf{F}_{k-1}$, $\mathbf{L}_{k-1}$, $\mathbf{H}_k$, and $\mathbf{M}_k$ are called the *Jacobian matrices* of the system

EKF | Computing Jacobian matrices

In vector calculus, a *Jacobian matrix* is the matrix of all first-order partial derivatives of a vector-valued function

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{f}}{\partial x_1} & \cdots & \frac{\partial \mathbf{f}}{\partial x_n} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

Intuitively, the Jacobian matrix tells you how fast each output of the function is changing along each input dimension

For example:

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ x_1^2 \end{bmatrix} \rightarrow \frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 2x_1 & 0 \end{bmatrix}$$

例如：

$f(x, y) = (x^4 + 3y^2 x,\ 5y^2 - 2xy + 1)$

$\frac{\partial f_1}{\partial x} = 4x^3 + 3y^2 \quad \frac{\partial f_1}{\partial y} = 6yx$

$\frac{\partial f_2}{\partial x} = -2y \quad \frac{\partial f_2}{\partial y} = 10 - 2x$

$$J_{f(x,y)} = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{pmatrix} = \begin{pmatrix} 4x^3 + 3y^2 & 6yx \\ -2y & 10 - 2x \end{pmatrix}$$

例3：

$$J(x,y,z) = \left(x e^{2y} \cos(-z),\ (y-2)^3 \cdot \sin\left(\frac{z}{2}\right),\ e^{2y} \cdot \ln\left(\frac{x}{3}\right) \right)$$

$$\begin{pmatrix} \frac{\partial f_1}{\partial x}, & \frac{\partial f_1}{\partial y}, & \frac{\partial f_1}{\partial z}, \\ \frac{\partial f_2}{\partial x}, & \frac{\partial f_2}{\partial y}, & \frac{\partial f_2}{\partial z} \\ \frac{\partial f_3}{\partial x}, & \frac{\partial f_3}{\partial y}, & \frac{\partial f_3}{\partial z}, \end{pmatrix}$$

$$\frac{\partial f_1}{\partial x} = e^{2y} \cos(-z)$$

$$\frac{\partial f_1}{\partial y} = 2e^{2y} \times \cos(-z)$$

$$\frac{\partial f_1}{\partial z} = \sin(-z)\, e^{2y} \times .$$

⋮

EKF | Putting it all together

With our linearized models and Jacobians, we can now use the Kalman Filter equations!

Linearized motion model

$$\mathbf{x}_k = \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}) + \mathbf{F}_{k-1}\left(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}\right) + \mathbf{L}_{k-1}\mathbf{w}_{k-1}$$

Linearized measurement model

$$\mathbf{y}_k = \mathbf{h}_k(\check{\mathbf{x}}_k, \mathbf{0}) + \mathbf{H}_k\left(\mathbf{x}_k - \check{\mathbf{x}}_k\right) + \mathbf{M}_k\mathbf{v}_k$$

Prediction

$$\check{\mathbf{x}}_k = \mathbf{f}_{k-1}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0})$$
$$\check{\mathbf{P}}_k = \mathbf{F}_{k-1}\hat{\mathbf{P}}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{L}_{k-1}\mathbf{Q}_{k-1}\mathbf{L}_{k-1}^T$$

Optimal gain

$$\mathbf{K}_k = \check{\mathbf{P}}_k\mathbf{H}_k^T(\mathbf{H}_k\check{\mathbf{P}}_k\mathbf{H}_k^T + \mathbf{M}_k\mathbf{R}_k\mathbf{M}_k^T)^{-1}$$

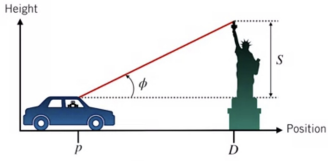
$\check{\mathbf{x}}_k$ Prediction (given motion model) at time k

$\hat{\mathbf{x}}_k$ Corrected prediction (given measurement) at time k

Correction

$$\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{y}_k - \mathbf{h}_k(\check{\mathbf{x}}_k, \mathbf{0}))$$
$$\hat{\mathbf{P}}_k = (\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)\check{\mathbf{P}}_k$$

EKF | Short example



$$\mathbf{x} = \begin{bmatrix} p \\ \dot{p} \end{bmatrix} \qquad \mathbf{u} = \ddot{p}$$

S and *D* are known in advance

Motion/Process model

$$\mathbf{x}_k = \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1})$$
$$= \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}\mathbf{x}_{k-1} + \begin{bmatrix} 0 \\ \Delta t \end{bmatrix}\mathbf{u}_{k-1} + \mathbf{w}_{k-1}$$

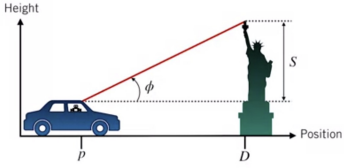
Landmark measurement model

$$y_k = \phi_k = h(p_k, v_k)$$
$$= \tan^{-1}\left(\frac{S}{D - p_k}\right) + v_k$$

Noise densities

$$v_k \sim \mathcal{N}(0,\ 0.01) \qquad \mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, (0.1)\mathbf{1}_{2\times 2})$$

EKF | Short example



Motion model Jacobians

$$\mathbf{F}_{k-1} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}_{k-1}}\right|_{\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{L}_{k-1} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{w}_{k-1}}\right|_{\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0}} = \mathbf{1}_{2\times 2}$$

Measurement model Jacobians

$$\mathbf{H}_k = \left.\frac{\partial h}{\partial \mathbf{x}_k}\right|_{\check{\mathbf{x}}_k, \mathbf{0}} = \begin{bmatrix} \frac{S}{(D - \check{p}_k)^2 + S^2} & 0 \end{bmatrix}$$

$$M_k = \left.\frac{\partial h}{\partial v_k}\right|_{\check{\mathbf{x}}_k, \mathbf{0}} = 1$$

Data

$$\hat{\mathbf{x}}_0 \sim \mathcal{N}\left(\begin{bmatrix} 0 \\ 5 \end{bmatrix}, \begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix}\right)$$

$\Delta t = 0.5$ s

$u_0 = -2\ [m/s^2]$ $\qquad y_1 = \pi/6\ [\text{rad}]$

$S = 20\ [m]$ $\qquad D = 40\ [m]$

Prediction

$$\check{\mathbf{x}}_1 = \mathbf{f}_0(\hat{\mathbf{x}}_0, \mathbf{u}_0, \mathbf{0})$$
$$\begin{bmatrix} \check{p}_1 \\ \check{\dot{p}}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 5 \end{bmatrix} + \begin{bmatrix} 0 \\ 0.5 \end{bmatrix}(-2) = \begin{bmatrix} 2.5 \\ 4 \end{bmatrix}$$

$$\check{\mathbf{P}}_1 = \mathbf{F}_0\hat{\mathbf{P}}_0\mathbf{F}_0^T + \mathbf{L}_0\mathbf{Q}_0\mathbf{L}_0^T$$
$$\check{\mathbf{P}}_1 = \begin{bmatrix} 1 & 0.5 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0.01 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0.5 & 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.36 & 0.5 \\ 0.5 & 1.1 \end{bmatrix}$$

Kalman Gain.

Correction

$$\mathbf{K}_1 = \check{\mathbf{P}}_1\mathbf{H}_1^T(\mathbf{H}_1\check{\mathbf{P}}_1\mathbf{H}_1^T + \mathbf{M}_1\mathbf{R}_1\mathbf{M}_1^T)^{-1}$$
$$= \begin{bmatrix} 0.36 & 0.5 \\ 0.5 & 1.1 \end{bmatrix}\begin{bmatrix} 0.011 \\ 0 \end{bmatrix}\left(\begin{bmatrix} 0.011 & 0 \end{bmatrix}\begin{bmatrix} 0.36 & 0.5 \\ 0.5 & 1.1 \end{bmatrix}\begin{bmatrix} 0.011 \\ 0 \end{bmatrix} + 1(0.01)(1)\right)^{-1}$$
$$= \begin{bmatrix} 0.40 \\ 0.55 \end{bmatrix}$$

$$\hat{\mathbf{x}}_1 = \check{\mathbf{x}}_1 + \mathbf{K}_1(\mathbf{y}_1 - \mathbf{h}_1(\check{\mathbf{x}}_1, \mathbf{0}))$$
$$\begin{bmatrix} \hat{p}_1 \\ \hat{\dot{p}}_1 \end{bmatrix} = \begin{bmatrix} 2.5 \\ 4 \end{bmatrix} + \begin{bmatrix} 0.40 \\ 0.55 \end{bmatrix}(0.52 - 0.49) = \begin{bmatrix} 2.51 \\ 4.02 \end{bmatrix}$$

Bonus!

$$\hat{\mathbf{P}}_1 = (\mathbf{1} - \mathbf{K}_1\mathbf{H}_1)\check{\mathbf{P}}_1$$
$$= \begin{bmatrix} 0.36 & 0.50 \\ 0.50 & 1.1 \end{bmatrix}$$

延伸.

Error State - EKF.

$$\mathbf{x} = \hat{\mathbf{x}} + \delta\mathbf{x}$$

True State — Nominal State ("Large") — Error State ("Small")

1. Update nominal state with motion model

$$\check{\mathbf{x}}_k = \mathbf{f}_{k-1}(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{0})$$

This could be $\check{\mathbf{x}}_{k-1}$ or $\hat{\mathbf{x}}_{k-1}$

2. Propagate uncertainty

$$\check{\mathbf{P}}_k = \mathbf{F}_{k-1}\mathbf{P}_{k-1}\mathbf{F}_{k-1}^T + \mathbf{L}_{k-1}\mathbf{Q}_{k-1}\mathbf{L}_{k-1}^T$$

This could be $\check{\mathbf{P}}_{k-1}$ or $\hat{\mathbf{P}}_{k-1}$

3. If a measurement is available:
   1. Compute Kalman Gain

$$\mathbf{K}_k = \check{\mathbf{P}}_k\mathbf{H}_k^T(\mathbf{H}_k\check{\mathbf{P}}_k\mathbf{H}_k^T + \mathbf{R})^{-1}$$

   2. Compute error state

$$\delta\hat{\mathbf{x}}_k = \mathbf{K}_k(\mathbf{y}_k - \mathbf{h}_k(\check{\mathbf{x}}_k, \mathbf{0}))$$

   3. Correct nominal state

$$\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k + \delta\hat{\mathbf{x}}_k$$

   4. Correct state covariance

$$\hat{\mathbf{P}}_k = (\mathbf{1} - \mathbf{K}_k\mathbf{H}_k)\check{\mathbf{P}}_k$$

Project 1#

Motion Model

The vehicle motion model recieves linear and angular velocity odometry readings as inputs, and outputs the state (i.e., the 2D pose) of the vehicle:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + T \begin{bmatrix} \cos\theta_{k-1} & 0 \\ \sin\theta_{k-1} & 0 \\ 0 & 1 \end{bmatrix} \left(\begin{bmatrix} v_k \\ \omega_k \end{bmatrix} + \mathbf{w}_k \right), \quad \mathbf{w}_k = \mathcal{N}(\mathbf{0}, \mathbf{Q})$$

- $\mathbf{x}_k = [x\ y\ \theta]^T$ is the current 2D pose of the vehicle
- v_k and ω_k are the linear and angular velocity odometry readings, which we use as inputs to the model

The process noise $\mathbf{w}_k$ has a (zero mean) normal distribution with a constant covariance $\mathbf{Q}$.

Measurement Model

The measurement model relates the current pose of the vehicle to the LIDAR range and bearing measurements $\mathbf{y}_k^l = [r\ \phi]^T$.

$$\mathbf{y}_k^l = \begin{bmatrix} \sqrt{(x_l - x_k - d\cos\theta_k)^2 + (y_l - y_k - d\sin\theta_k)^2} \\ atan2\,(y_l - y_k - d\sin\theta_k, x_l - x_k - d\cos\theta_k) - \theta_k \end{bmatrix} + \mathbf{n}_k^l, \quad \mathbf{n}_k^l = \mathcal{N}(\mathbf{0}, \mathbf{R})$$

- x_l and y_l are the ground truth coordinates of the landmark l
- x_k and y_k and θ_k represent the current pose of the vehicle
- d is the known distance between robot center and laser rangefinder (LIDAR)

The landmark measurement noise $\mathbf{n}_k^l$ has a (zero mean) normal distribution with a constant covariance $\mathbf{R}$.

$y = \begin{bmatrix} r \\ \phi \end{bmatrix}$

- Compute the measurement model Jacobians at $\check{\mathbf{x}}_k$

$$\mathbf{y}_k^l = \mathbf{h}(\mathbf{x}_k, \mathbf{n}_k^l)$$

$$\mathbf{H}_k = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}_k}\right|_{\check{\mathbf{x}}_k, 0}, \quad \mathbf{M}_k = \left.\frac{\partial \mathbf{h}}{\partial \mathbf{n}_k}\right|_{\check{\mathbf{x}}_k, 0}.$$

- Compute the Kalman Gain

$$H_k = \begin{bmatrix} \frac{\partial r}{\partial x_k} & \frac{\partial r}{\partial y_k} & \frac{\partial r}{\partial \theta_k} \\ \frac{\partial \phi}{\partial x_k} & \frac{\partial \phi}{\partial y_k} & \frac{\partial \phi}{\partial \theta_k} \end{bmatrix}$$

$$\frac{\partial r}{\partial x_k} = \frac{x_k + d\cos(\theta_k) - x_l}{r}$$