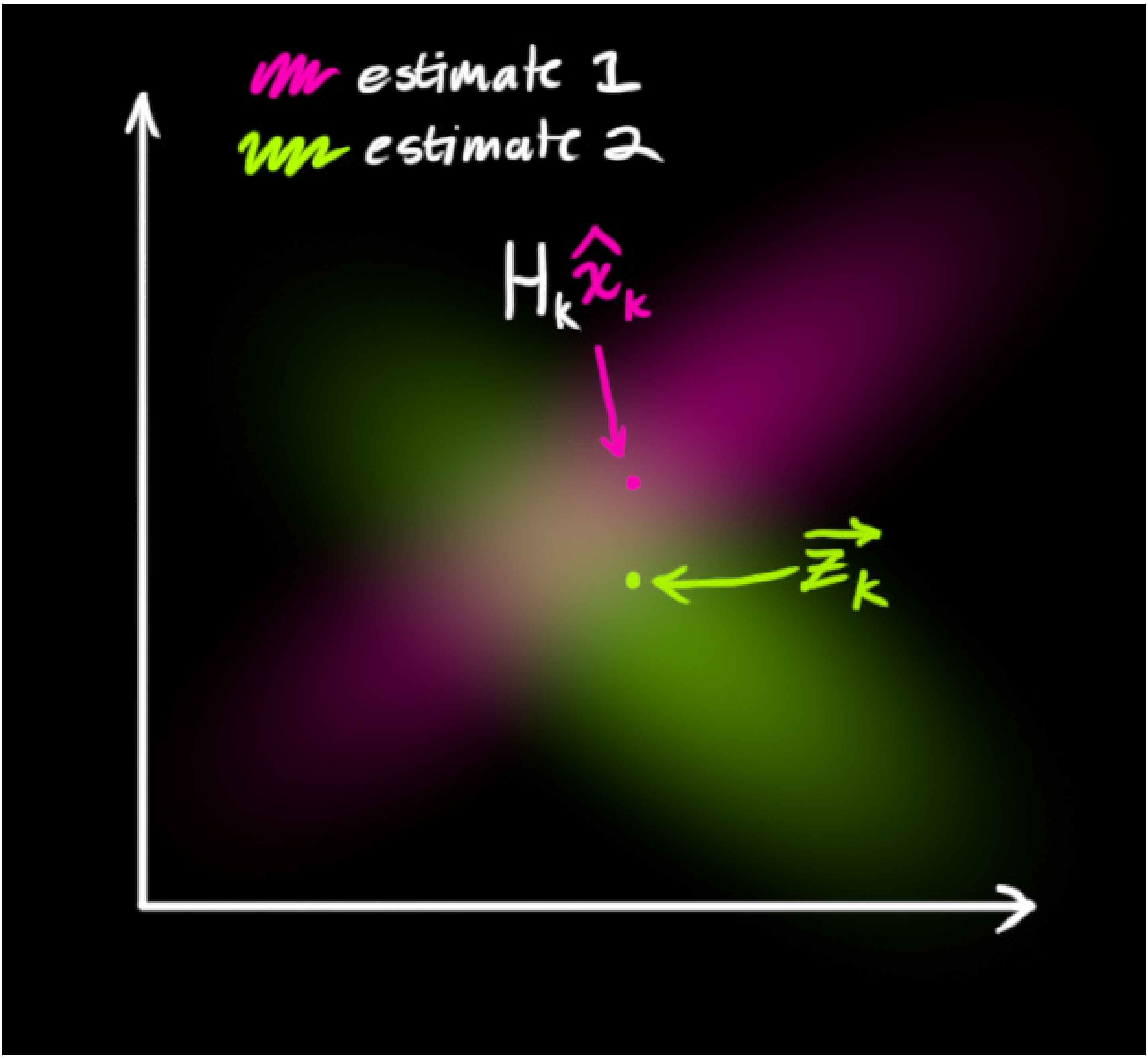


图说卡尔曼滤波，一份通俗易懂的教程

论智
调调参，论论AI 【公众号：论智 (jqr_AI) 】

作者：Bzarg

编译：Bot

编者按：卡尔曼滤波（Kalman filter）是一种高效的自回归滤波器，它能在存在诸多不确定性
的组合信息中估计动态系统的状态，是一种强大的、通用性极强的
尔曼，在一次访问NASA埃姆斯研究中心时，发现这种方法能帮助

本文是国外博主Bzarg在2015年写的一篇图解。虽然是几年前的文章，但是动态定位、自动导航、时间序列模型、卫星导航——卡尔曼滤波的应用范围依然非常广。那么，作为软件工程师和机器学习工程师，你真的了解卡尔曼滤波吗？

什么是卡尔曼滤波？

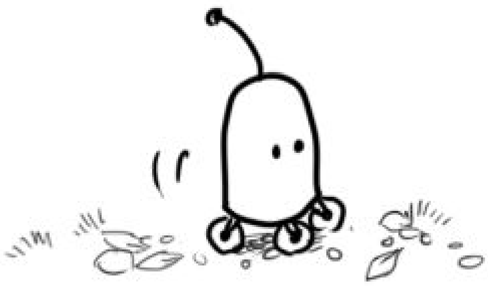
对于这个滤波器，我们几乎可以下这么一个定论：只要是存在不确定信息的动态系统，卡尔曼滤波就可以对系统下一步要做什么做出有根据的推测。即便有噪声信息干扰，卡尔曼滤波通常也能很好的弄清楚究竟发生了什么，找出现象间不易察觉的相关性。

因此卡尔曼滤波非常适合不断变化的系统，它的优点还有内存占用较小（只需保留前一个状态）、速度快，是实时问题和嵌入式系统的理想选择。

如果你曾经Google过卡尔曼滤波的教程（如今有一点点改善），你会发现相关的算法教程非常可怕，而且也没具体说清楚是什么。事实上，卡尔曼滤波很简单，如果我们以正确的方式看它，理解是很水到渠成的事。

本文会用大量清晰、美观的图片和颜色来解释这个概念，读者只需具备概率论和矩阵的一般基础知识。

我们能用卡尔曼滤波做什么？



也就是说，机器人有一个包含位置信息和速度信息的状态 $\vec{x}_k$ ：

$$\vec{x_k} = (\vec{p}, \vec{v})$$

注意，在这个例子中，状态是位置和速度，放进其他问题里，它也可以是水箱里的液体体积、汽车引擎温度、触摸板上指尖的位置，或者其他任何数据。

我们的小机器人装有GPS**传感器**，定位精度10米。虽然一般来说这点精度够用了，但我们希望它的定位误差能再小点，毕竟树林里到处都是土坑和陡坡，如果机器人稍稍偏了那么几米，它就有可能滚落山坡。所以GPS提供的信息还不够充分。

我们也可以**预测**机器人是怎么移动的：它会把指令发送给控制轮子的马达，如果这一刻它始终朝一个方向前进，没有遇到任何障碍物，那么下一刻它可能会继续坚持这个路线。但是机器人对自己的状态不是全知的：它可能会逆风行驶，轮子打滑，滚落颠簸地形……所以车轮转动次数并不能完全代表实际行驶距离，基于这个距离的预测也不完美。

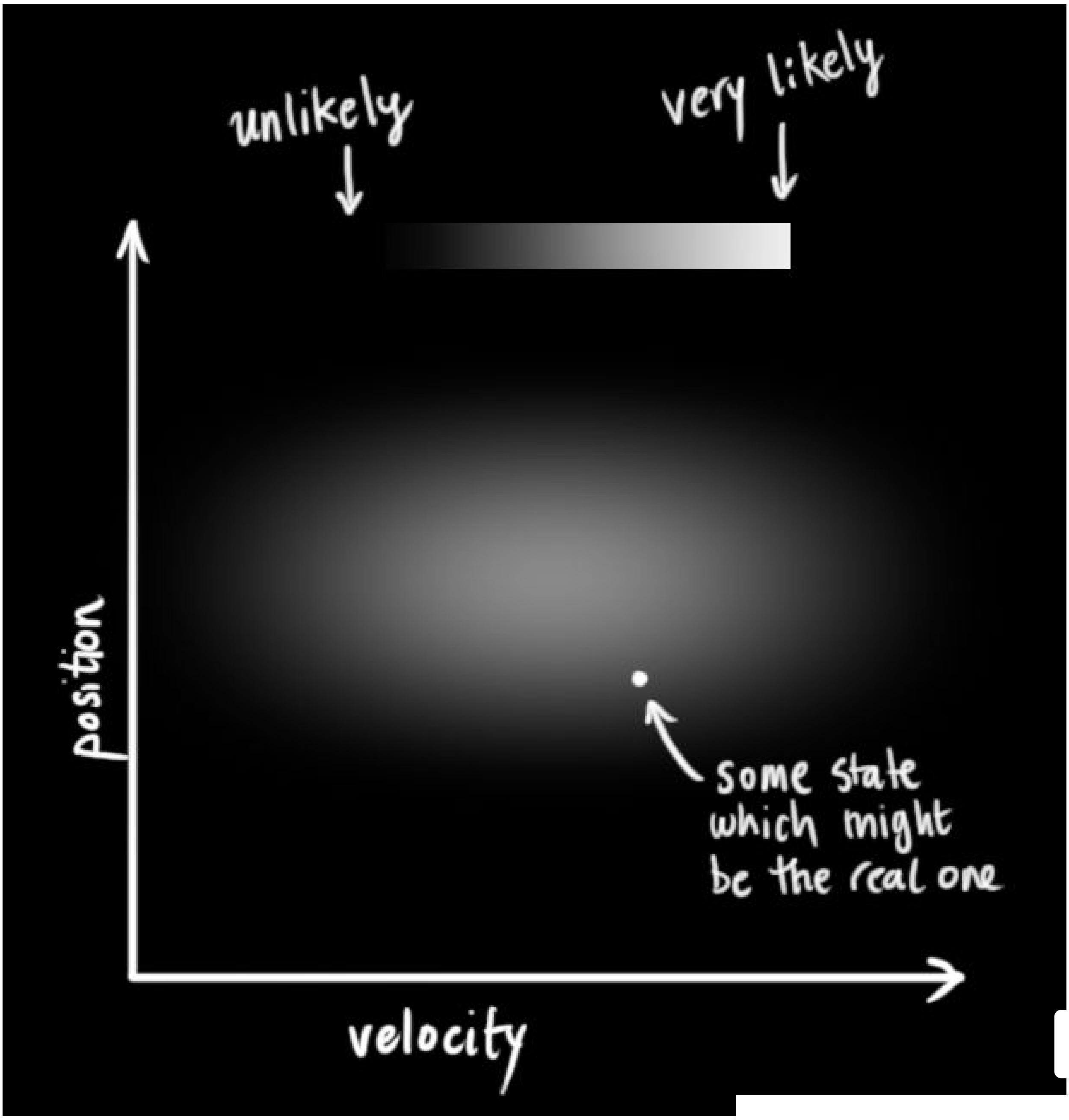
这个问题下，GPS为我们提供了一些关于状态的信息，但那是间接的、不准确的；我们的预测提供了关于机器人轨迹的信息，但那也是间接的、不准确的。

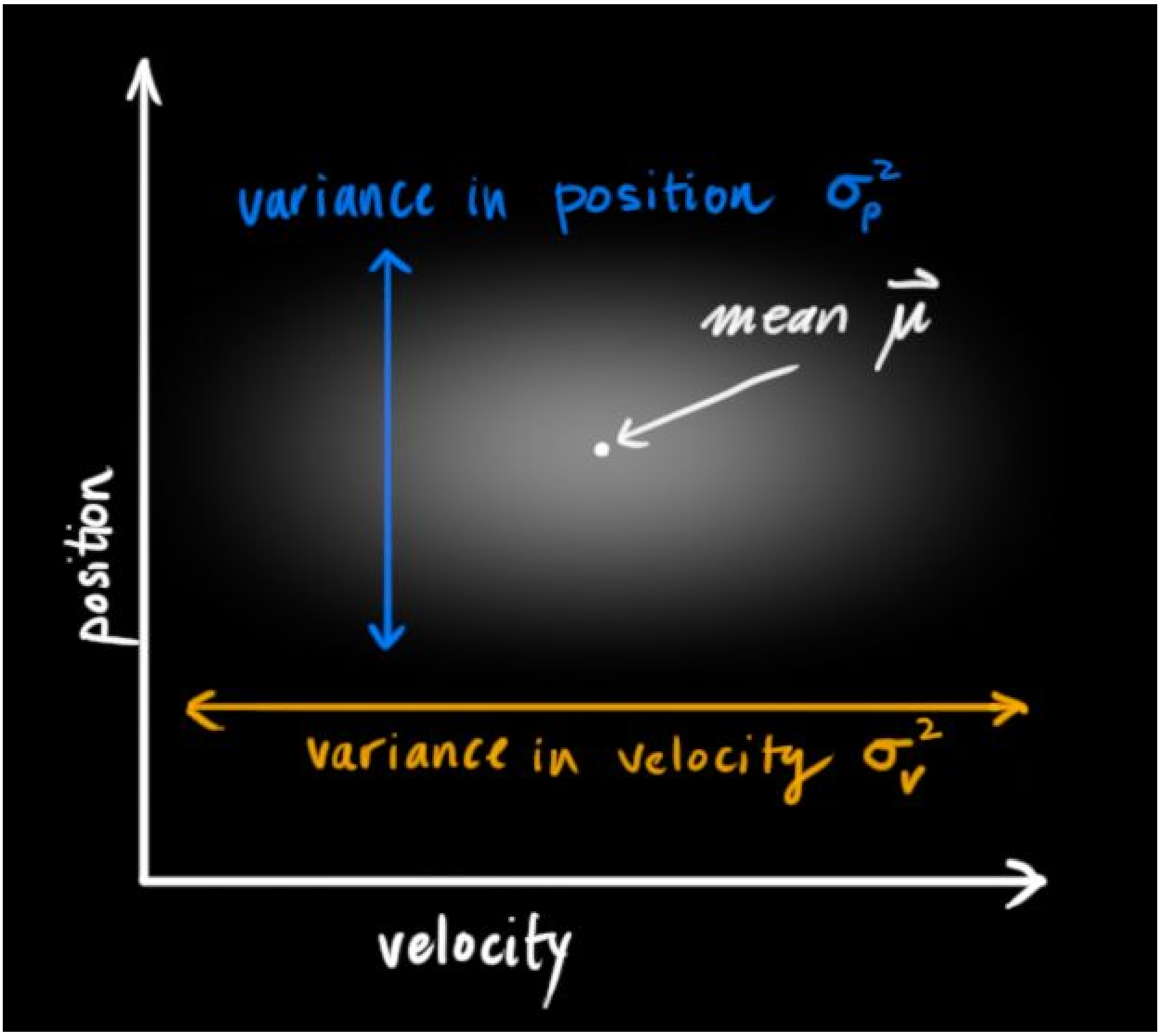
卡尔曼滤波眼里的机器人问题

还是上面这个问题，我们有一个状态，它和速度、位置有关：

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

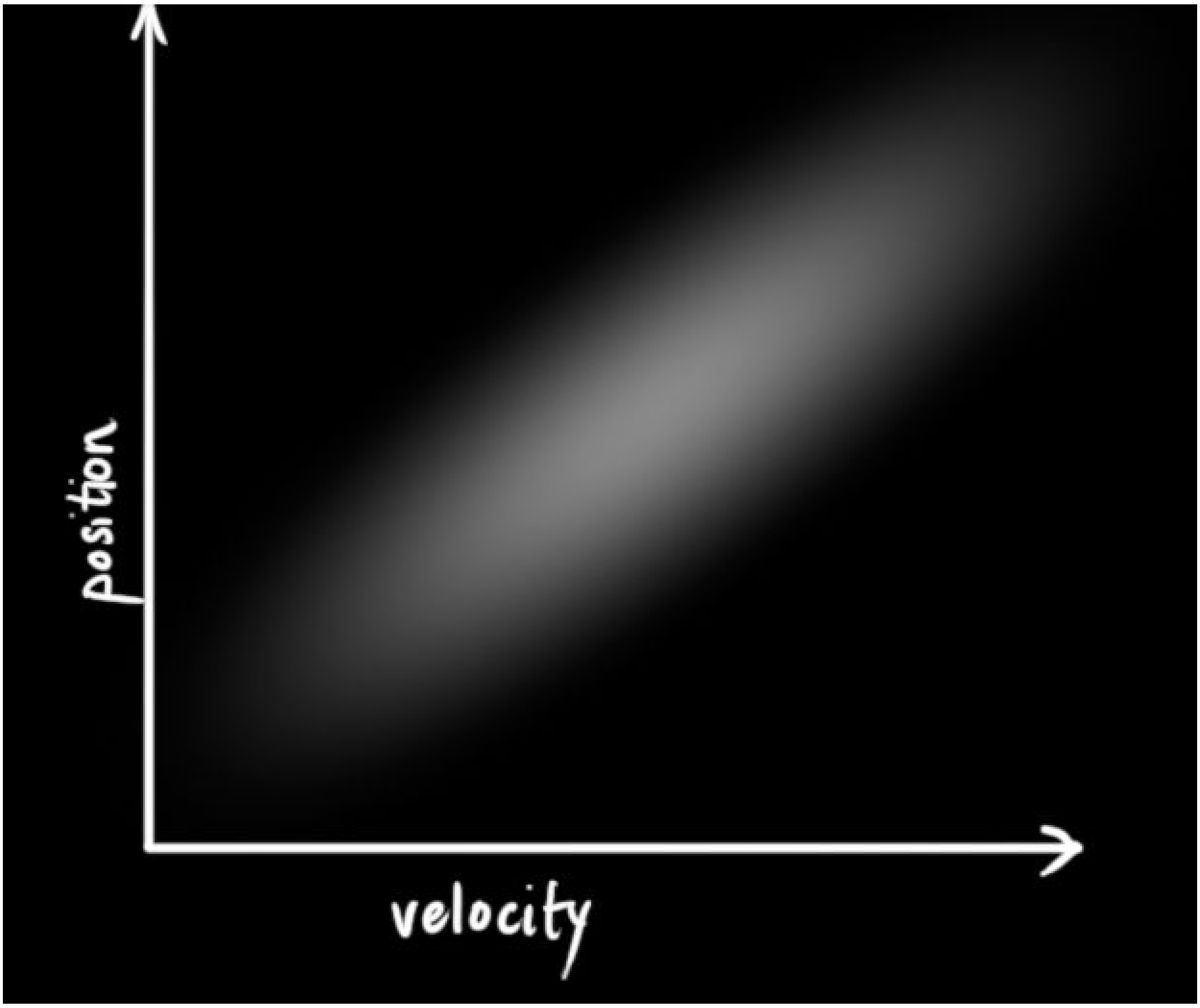
我们不知道它们的实际值是多少，但掌握着一些速度和位置的可能组合，其中某些组合的可能性更高：





在上图中，位置和速度是**不相关**的，这意味着我们不能从一个变量推测另一个变量。

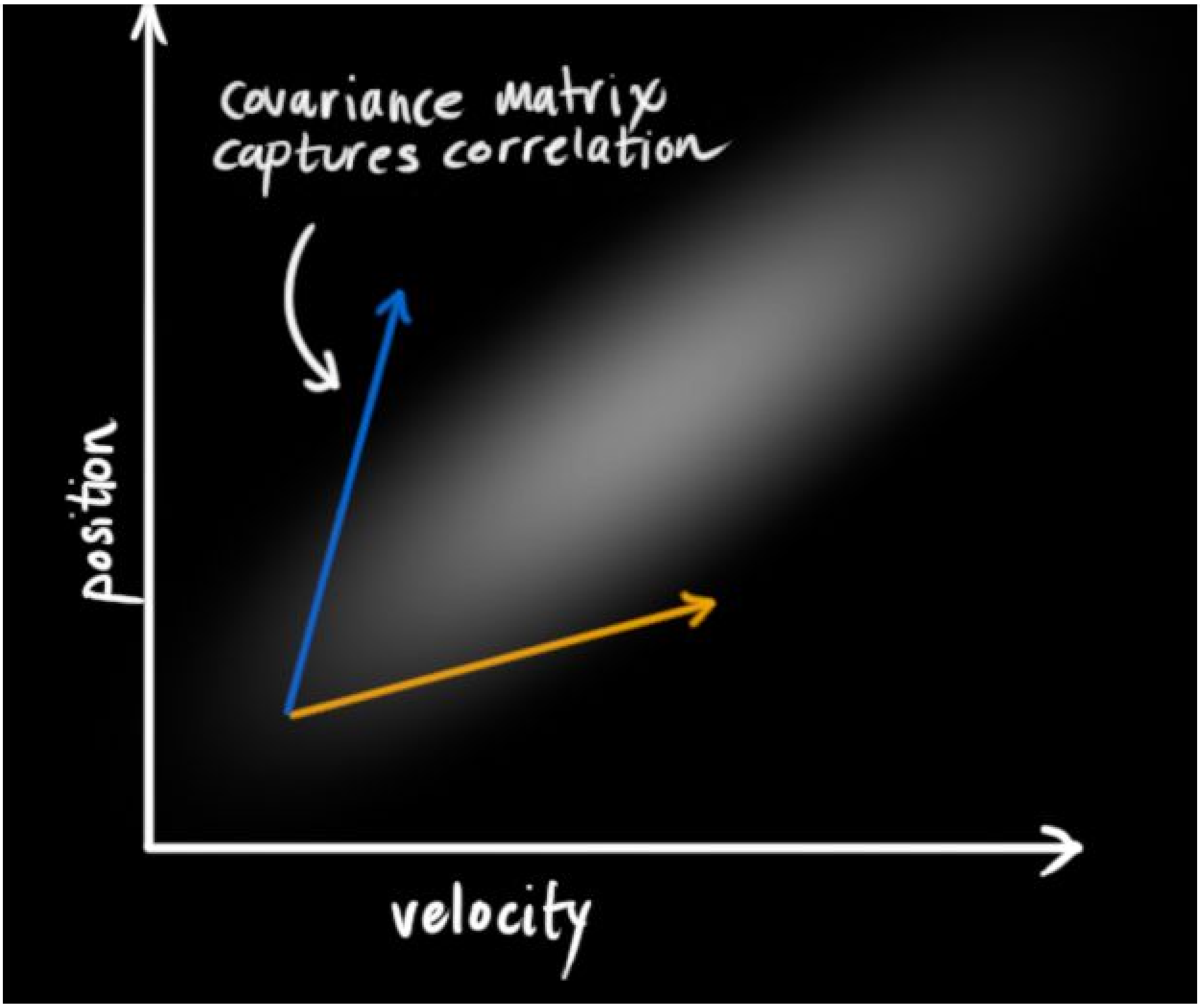
那么如果位置和速度相关呢？如下图所示，机器人前往特定位置的可能性取决于它拥有的速度。



这不难理解，如果基于旧位置估计新位置，我们会产生这两个结论：如果速度很快，机器人可能移动得更远，所以得到的位置会更远；如果速度很慢，机器人就走不了那么远。

这种关系对目标跟踪来说非常重要，因为它提供了更多信息：一个可以衡量可能性的标准。这就是卡尔曼滤波的目标：从不确定信息中挤出尽可能多的信息！

为了捕获这种相关性，我们用的是协方差矩阵。简而言之，矩阵的每个值是第 i 个变量和第 j 个变量之间的相关程度（由于矩阵是对称的， i 和 j 的位置可以随便交换）。我们用 Σ 表示协方差矩阵，在这个例子中，就是 Σ_{ij} 。

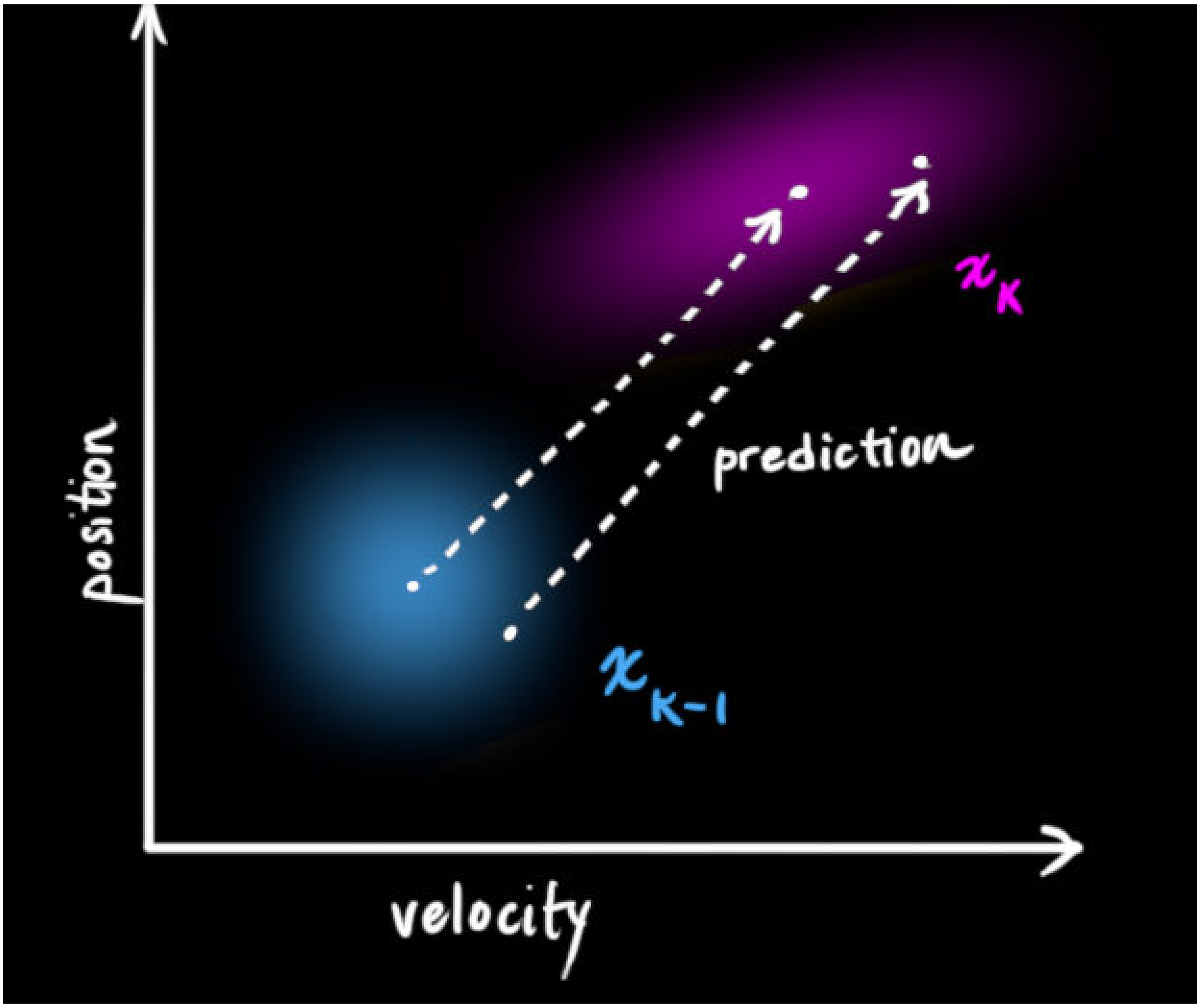


用矩阵描述问题

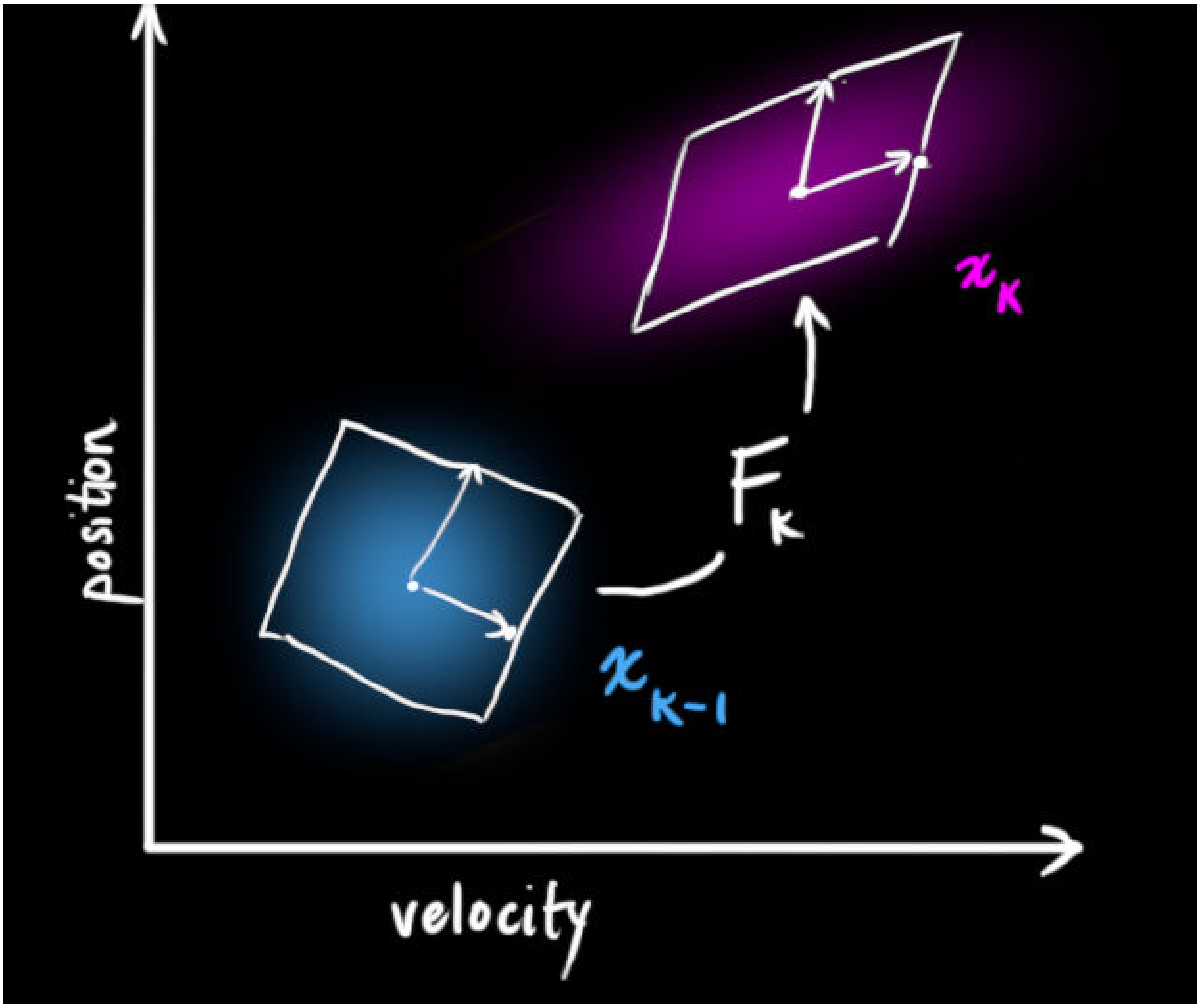
为了把以上关于状态的信息建模为高斯分布（图中色块），我们还需要 k 时的两个信息：最佳估计 $\hat{x}_k$ （均值，也就是 μ ），协方差矩阵 P_k 。（虽然还是用了位置和速度两个变量，但只要和问题相关，卡尔曼滤波可以包含任意数量的变量）

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

接下来，我们要通过查看当前状态（k-1时）来预测下一个状态（k时）。这里我们查看的状态不是真值，但预测函数无视真假，可以给出新分布：



我们可以用矩阵 F_k 表示这个预测步骤：



它从原始预测中取每一点，并将其移动到新的预测位置。如果原始预测是正确的，系统就会移动到新位置。

这是怎么做到的？为什么我们可以用矩阵来预测机器人下一刻的位置和速度？下面是个简单公式：

$$
\begin{aligned}
p_k &= p_{k-1} + \Delta t v_{k-1} \\
v_k &= \quad\quad\quad\;\; v_{k-1}
\end{aligned}
$$

换成矩阵形式：

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

这是一个预测矩阵，它能给出机器人的下一个状态，但目前我们还不知道协方差矩阵的更新方法。这也是我们要引出下面这个等式的原因：如果我们将分布中的每个

矩阵会发生什么变化

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

把这个式子和上面的最佳估计 $\hat{x}_k$ 结合，可得：

$$\begin{aligned} \hat{\mathbf{x}}_k &= \mathbf{F}_k \hat{\mathbf{x}}_{k-1} \\ \mathbf{P}_k &= \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T \end{aligned}$$

外部影响

但是，除了速度和位置，外因也会对系统造成影响。比如模拟火车运动，除了列车自驾系统，列车操作员可能会手动调速。在我们的机器人示例中，导航软件也可以发出停止指令。对于这些信息，我们把它作为一个向量 $\vec{u}_k$ ，纳入预测系统作为修正。

假设油门设置和控制命令是已知的，我们知道火车的预期加速度 a 。根据运动学基本定理，我们可得：

$$\begin{aligned} p_k &= p_{k-1} + \Delta t v_{k-1} + \frac{1}{2} a \Delta t^2 \\ v_k &= \qquad\qquad\quad v_{k-1} + a \Delta t \end{aligned}$$

把它转成矩阵形式：

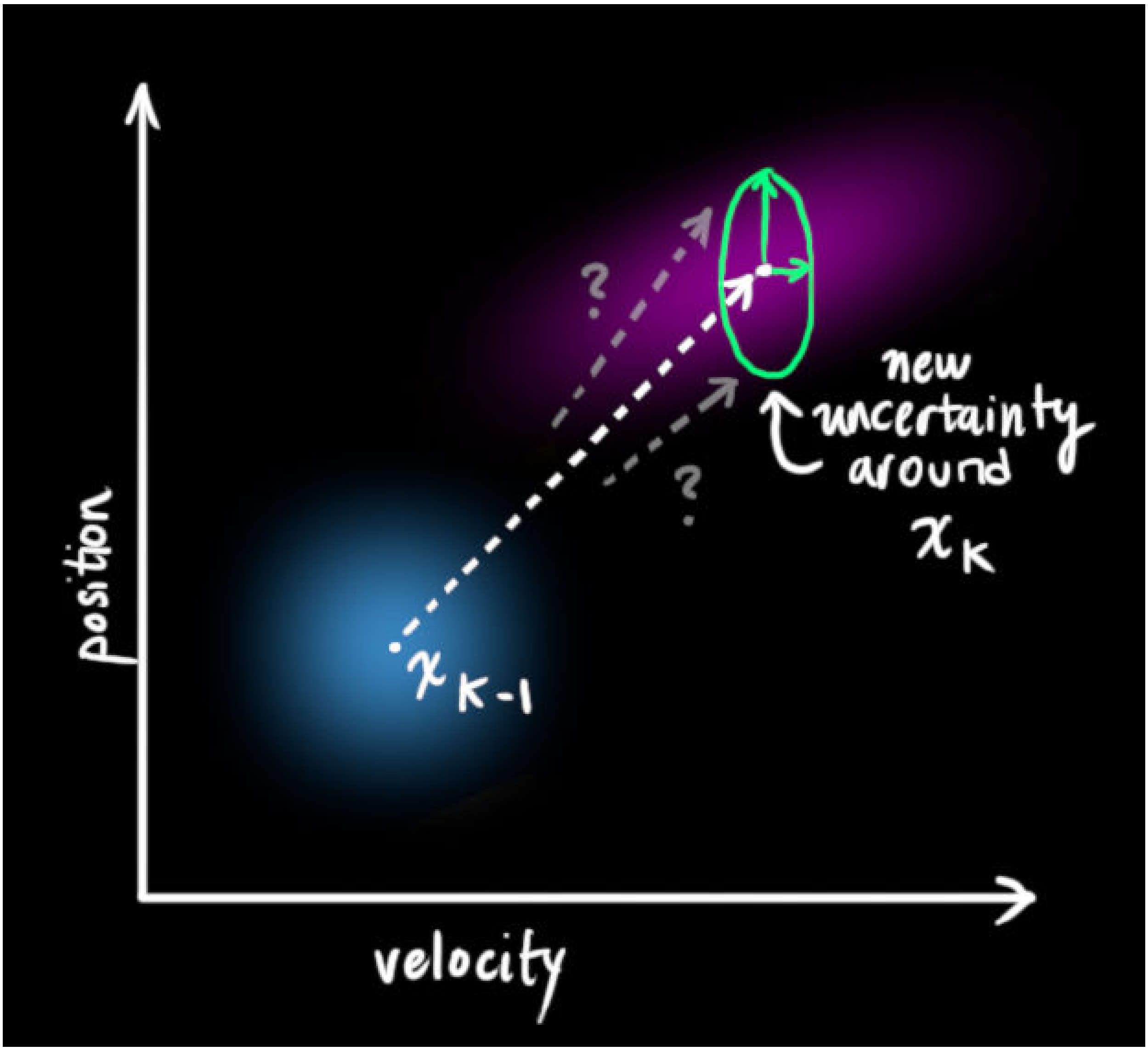
$$\begin{aligned} \hat{\mathbf{x}}_k &= \mathbf{F}_k \hat{\mathbf{x}}_{k-1} + \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix} a \\ &= \mathbf{F}_k \hat{\mathbf{x}}_{k-1} + \mathbf{B}_k \vec{\mathbf{u}}_k \end{aligned}$$

B_k 是控制矩阵， $\vec{u}_k$ 是控制向量。如果外部环境异常简单，我们可以忽略这部分内容，但是如果添加了外部影响后，模型的准确率还是上不去，这又是为什么呢？

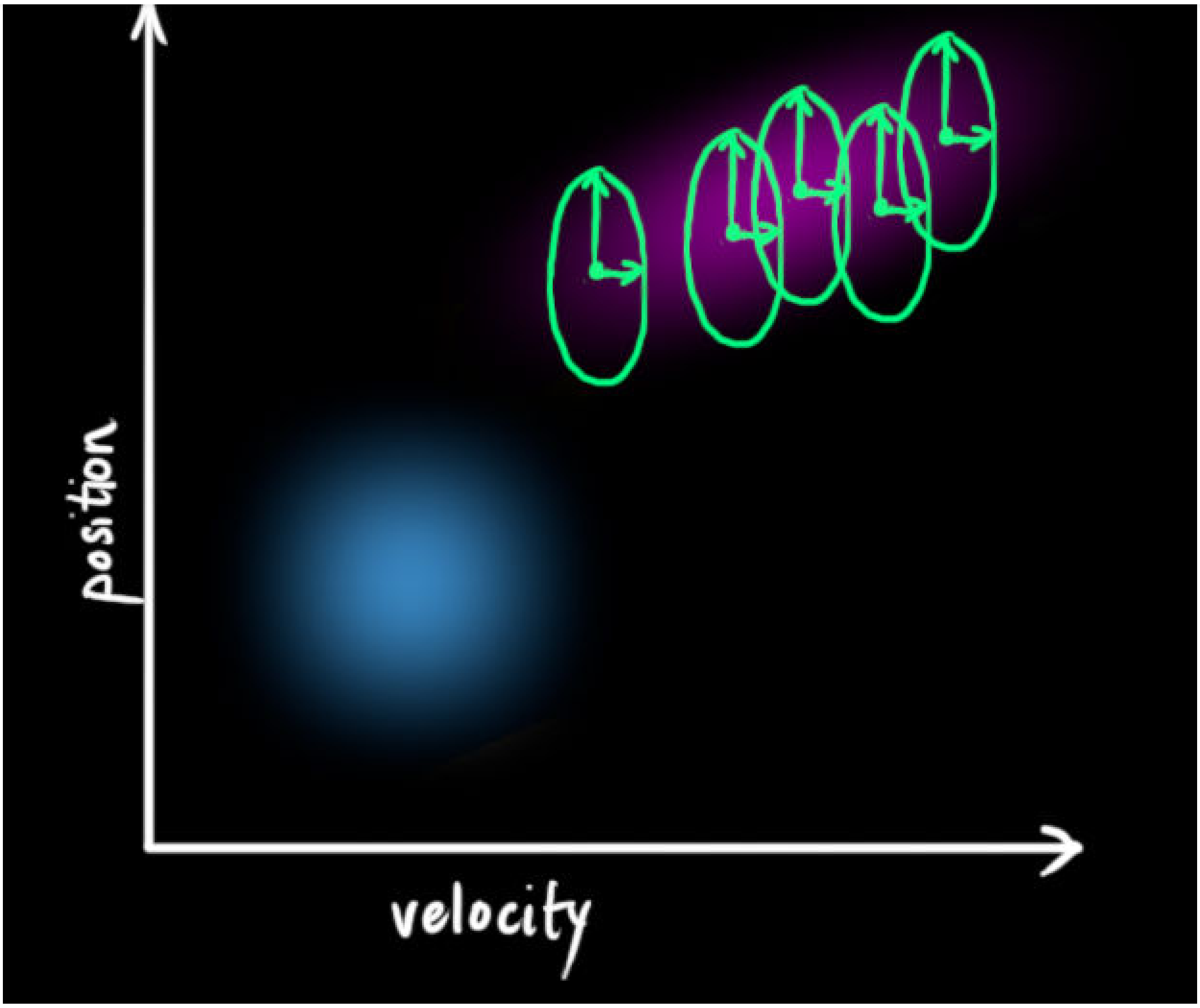
外部不确定性

当一个国家只按照自己的步子发展时，它会自生自灭。当一个国家开始依赖外部力量发展时，只要这些外部力量是已知的，我们也能预测它的存亡。

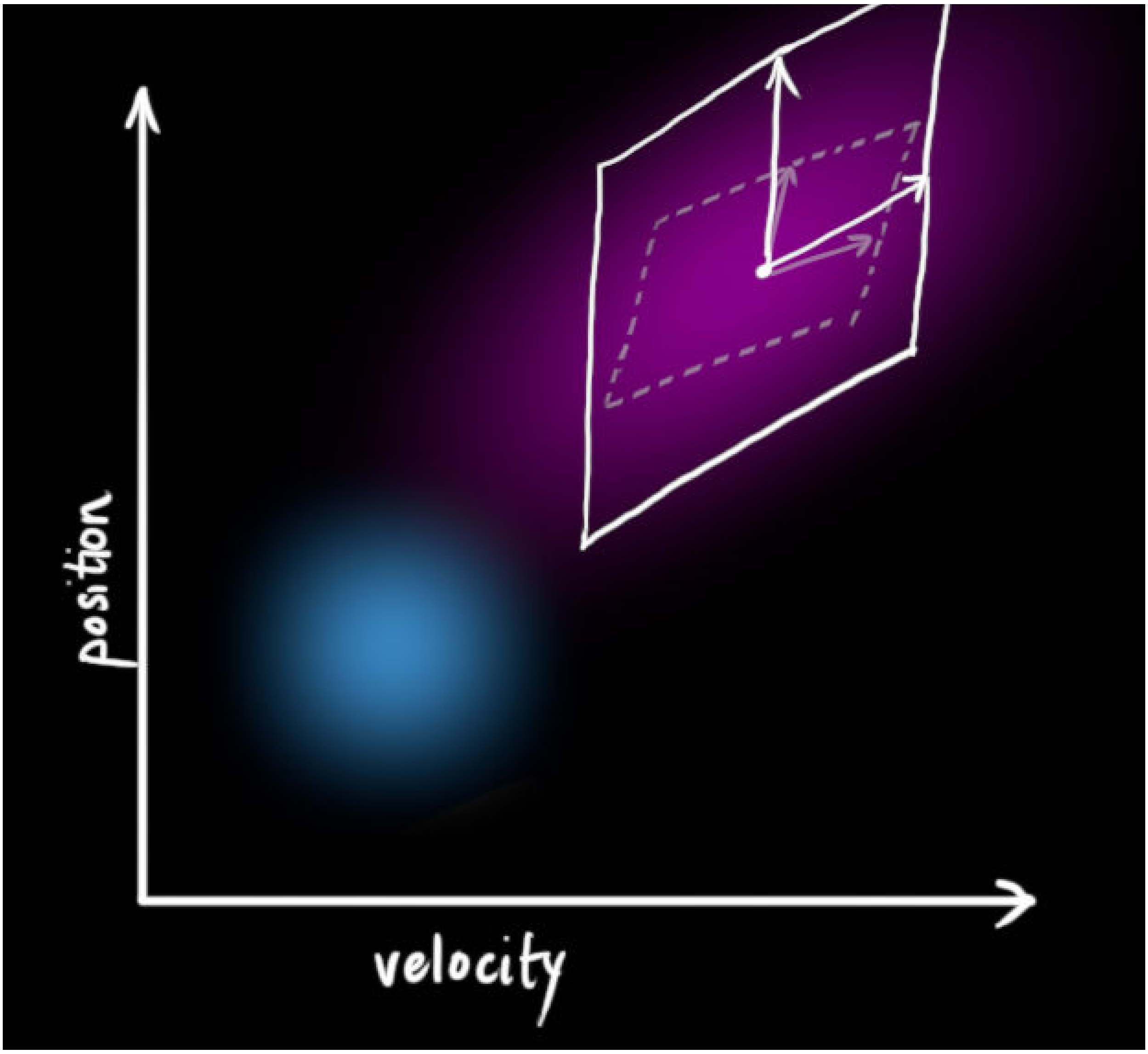
但是，如果存在我们不知道的力量呢？当我们监控无人机时，它可能会受到风的影响；当我们驾驶轮式机器人时，它的轮胎可能会打滑，或者粗糙地面会降低它的移速。这些因素是难以掌握的，如果出现其中的任意一种情况，预测结果就难以保障。



如上图所示，加上外部不确定性后， $\hat{x}_{k-1}$ 的每个预测状态都可能会移动到另一点，也就是蓝色的高斯分布会移动到紫色高斯分布的位置，并且具有协方差 Q_k 。换句话说，我们把这些不确定影响视为协方差 Q_k 的噪声。



这个紫色的高斯分布拥有和原分布相同的均值，但协方差不同。



我们在原式上加入 Q_k ：

$$
\begin{aligned}
\hat{\mathbf{x}}_k &= \mathbf{F}_k \hat{\mathbf{x}}_{k-1} + \mathbf{B}_k \vec{\mathbf{u}_k} \\
\mathbf{P}_k &= \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^T + \mathbf{Q}_k
\end{aligned}
$$

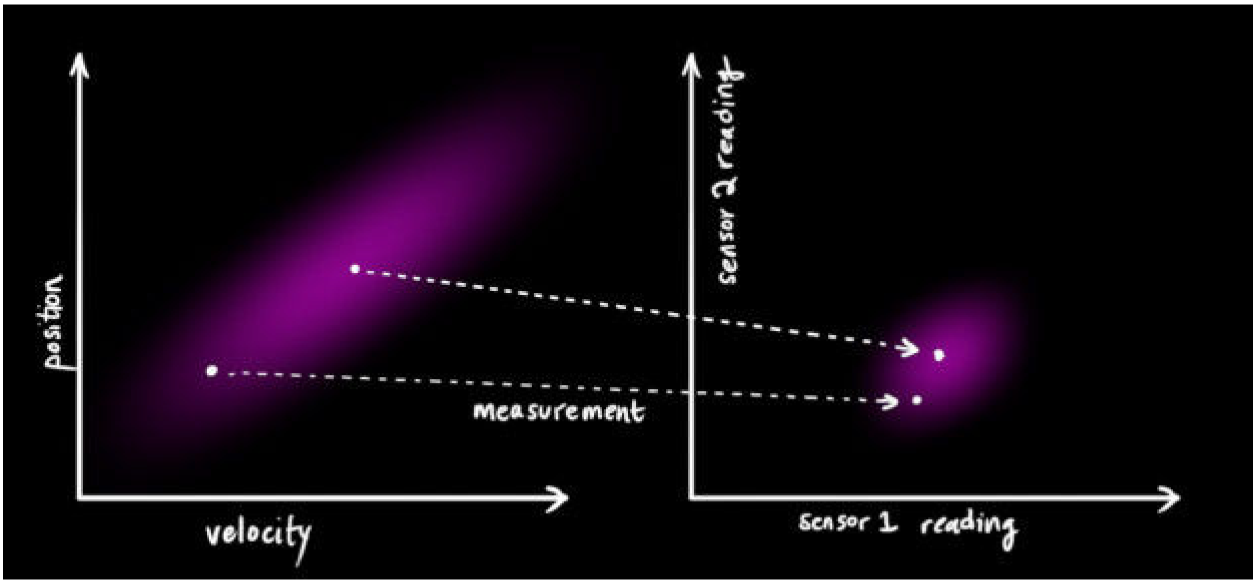
简而言之，这里：

新的最佳估计 是基于 原最佳估计 和 已知外部影响 校正后得到的预测。

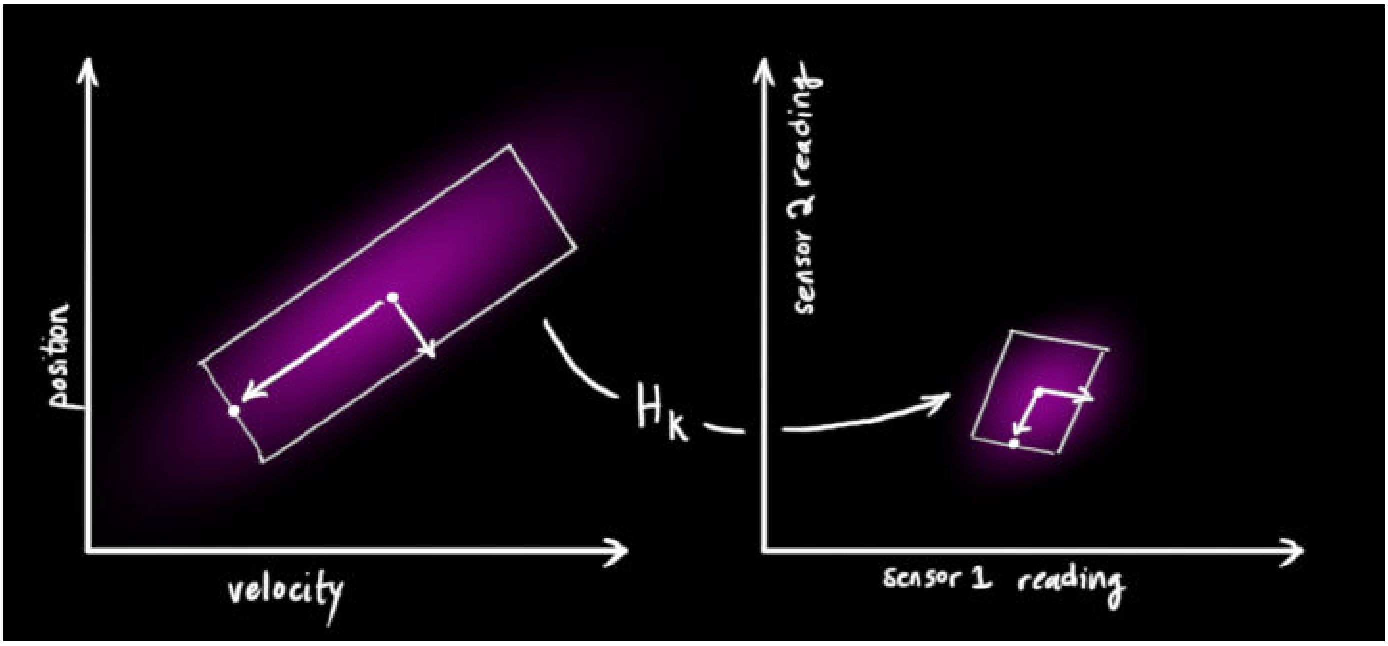
新的不确定性 是基于 原不确定性 和 外部环境的不确定性 得到的预测。

现在，有了这些概念介绍，我们可以把传感器数据输入其中。

我们可能有好几个传感器，它们一起提供有关系统状态的信息。传感器的作用不是我们关心的重点，它可以读取位置，可以读取速度，重点是，它能告诉我们关于状态的间接信息——它是状态下产生的一组读数。



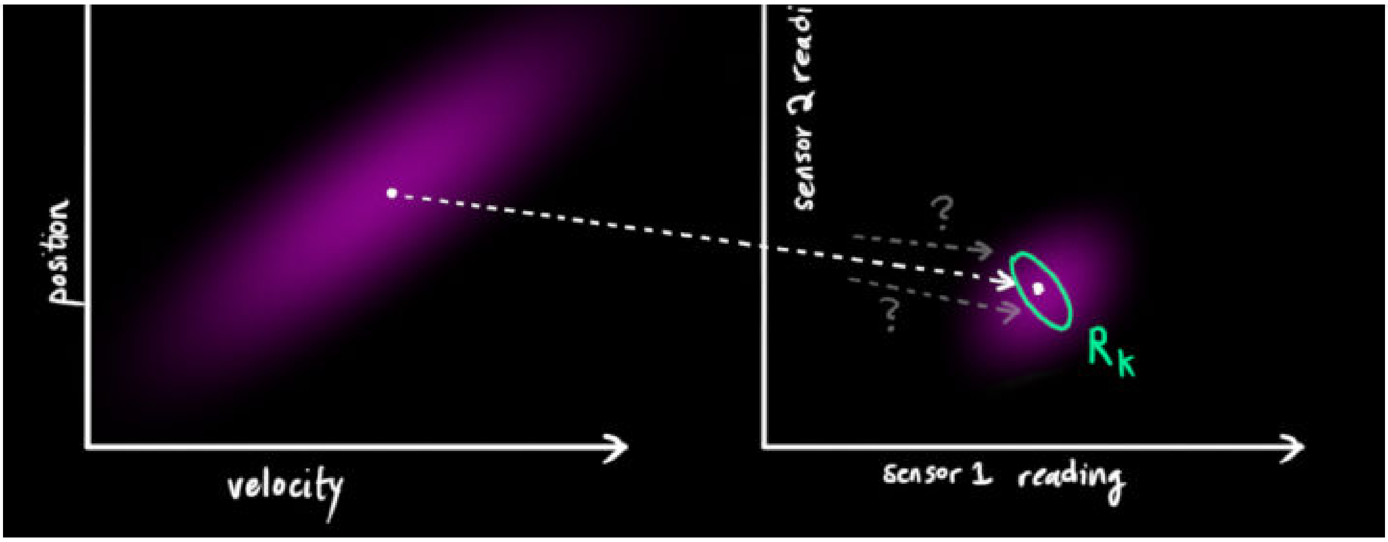
请注意，读数的规模和状态的规模不一定相同，所以我们把传感器读数矩阵设为 H_k 。



把这些分布转换为一般形式：

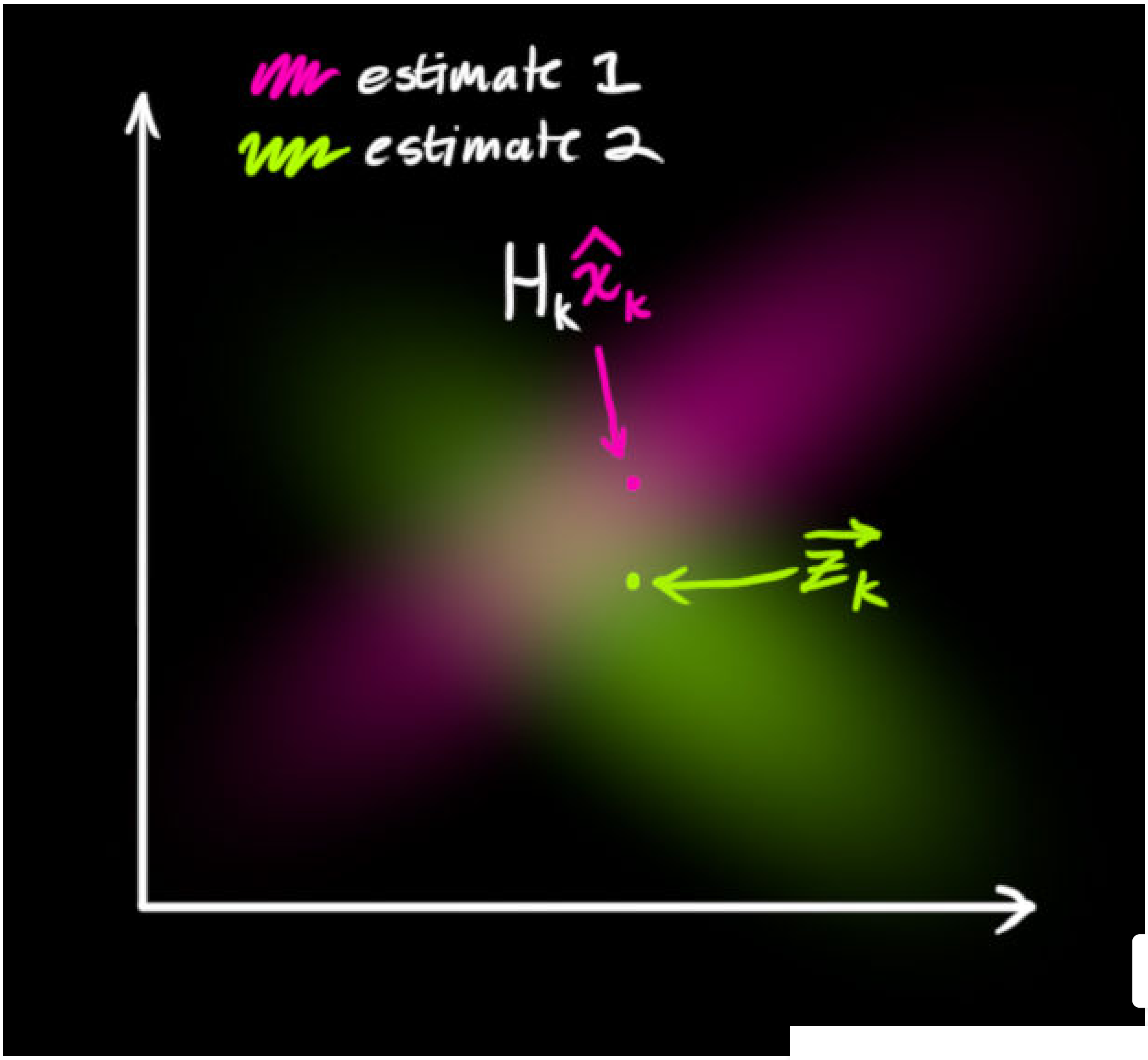
$$\vec{\mu}_{\text{expected}} = \mathbf{H}_k \hat{\mathbf{x}}_k$$
$$\mathbf{\Sigma}_{\text{expected}} = \mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T$$

卡尔曼滤波的一大优点是擅长处理传感器噪声。换句话说，由于种
是不准的，一个状态事实上可以产生多种读数。

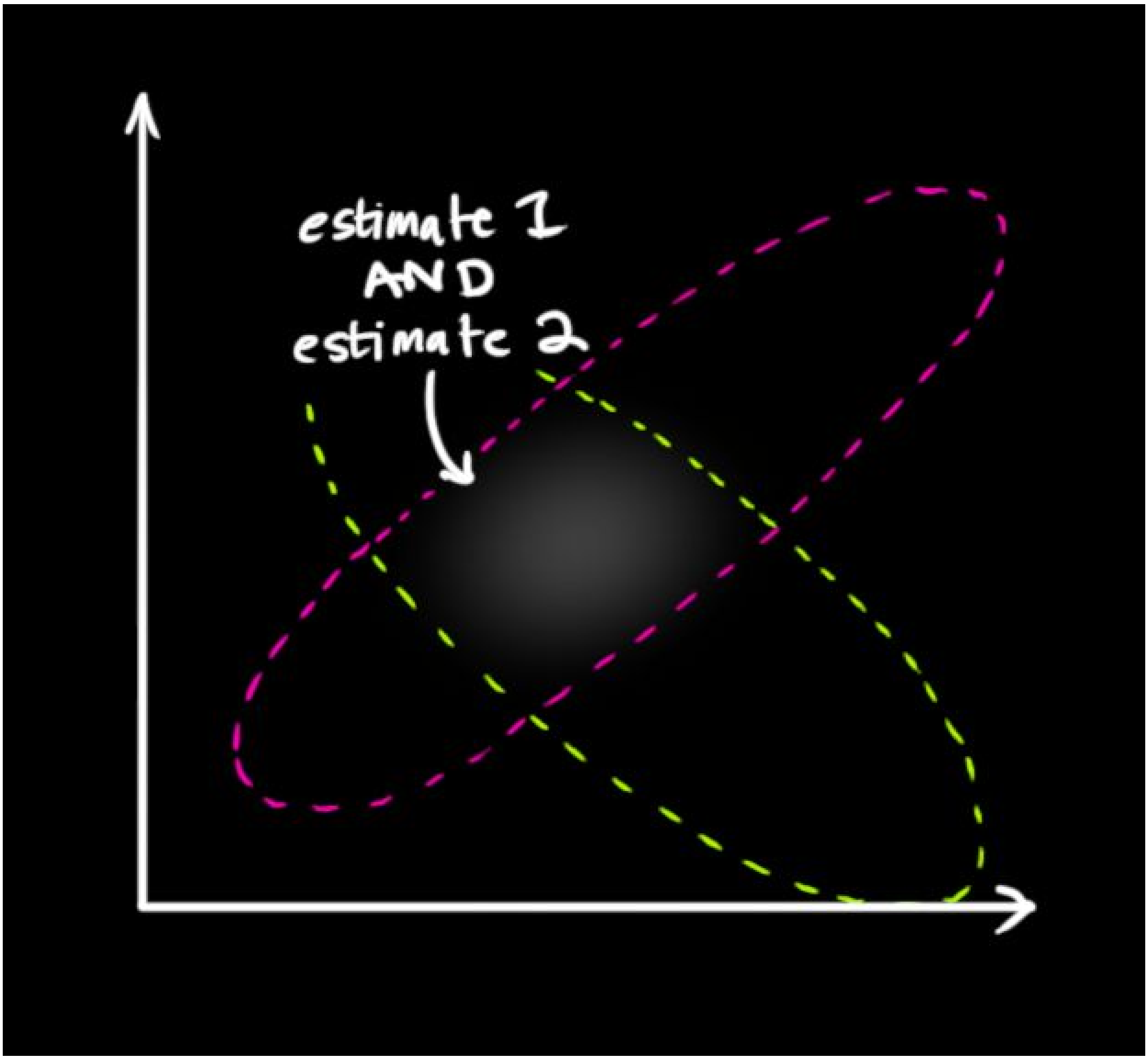


我们将这种不确定性（即传感器噪声）的协方差设为 R_k ，读数的分布均值设为 z_k 。

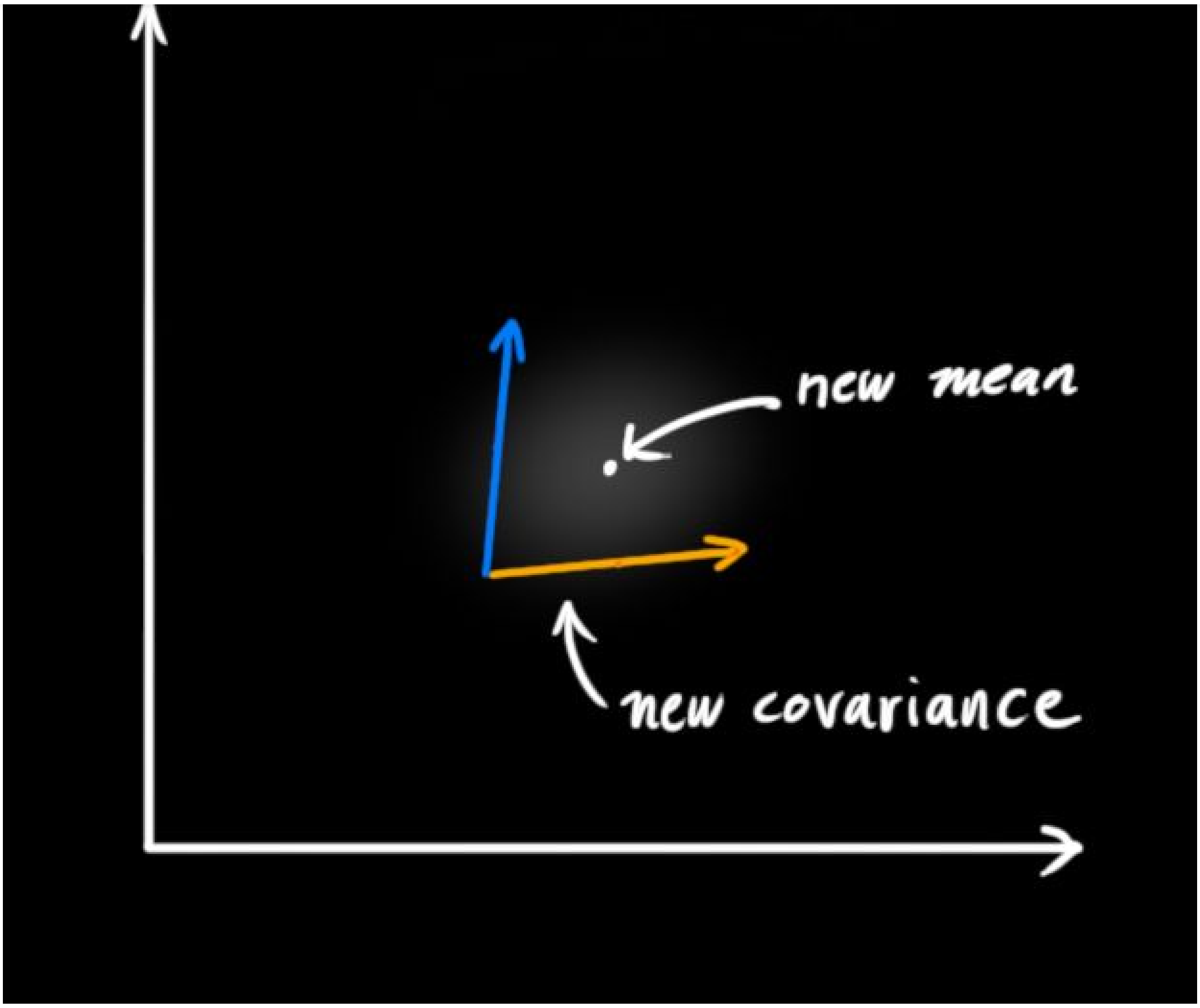
现在我们得到了两块高斯分布，一块围绕预测的均值，另一块围绕传感器读数。



最简单的方法是两者相乘：



两块高斯分布相乘后，我们可以得到它们的重叠部分，这也是会出现最佳估计的区域。换个角度看，它看起来也符合高斯分布：



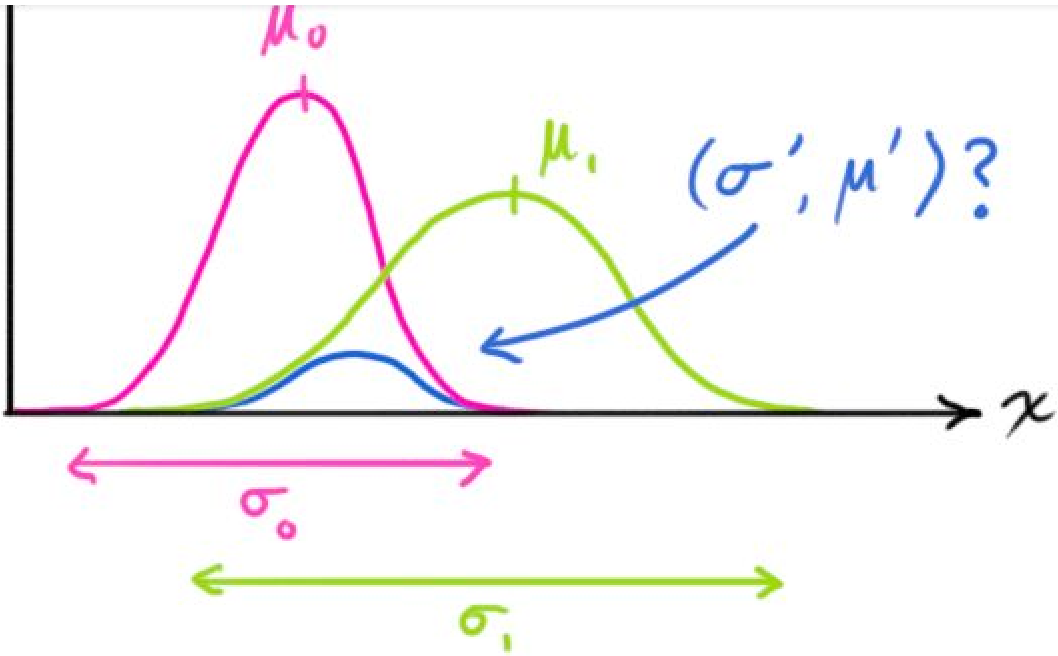
事实证明，当你把两个高斯分布和它们各自的均值和协方差矩阵相乘时，你会得到一个拥有独立均值和协方差矩阵的新高斯分布。最后剩下的问题就不难解决了：我们必须有一个公式来从旧的参数中获取这些新参数！

结合高斯

让我们从一维看起，设方差为 σ^2 ，均值为 μ ，一个标准一维高斯钟形曲线方程如下所示：

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

那么两条高斯曲线相乘呢？



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

把这个式子按照一维方程进行扩展，可得：

$$\mu' = \mu_0 + \frac{\sigma_0^2(\mu_1 - \mu_0)}{\sigma_0^2 + \sigma_1^2}$$
$$\sigma'^2 = \sigma_0^2 - \frac{\sigma_0^4}{\sigma_0^2 + \sigma_1^2}$$

如果有些太复杂，我们用k简化一下：

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

$$\mu' = \mu_0 + \mathbf{k}(\mu_1 - \mu_0)$$
$$\sigma'^2 = \sigma_0^2 - \mathbf{k}\sigma_0^2$$

以上是一维的内容，如果是多维空间，把这个式子转成矩阵格式：

$$\vec{\mu}' = \vec{\mu_0} + \mathbf{K}(\vec{\mu_1} - \vec{\mu_0})$$
$$\Sigma' = \Sigma_0 - \mathbf{K}\Sigma_0$$

这个矩阵 K 就是我们说的**卡尔曼增益**，easy！

把它们结合在一起

截至目前，我们有用矩阵 $(\mu_0, \Sigma_0) = (H_k \hat{x}_k, H_k P_k H_k^T)$ 预测的分布，有用传感器读数 $(\mu_1, \Sigma_1) = (\vec{z}_k, R_k)$ 预测的分布。把它们代入上节的矩阵等式中：

$$\begin{aligned}\mathbf{H}_k \hat{\mathbf{x}}_k' &= \mathbf{H}_k \hat{\mathbf{x}}_k & + \mathbf{K}(\vec{\mathbf{z}_k} - \mathbf{H}_k \hat{\mathbf{x}}_k) \\ \mathbf{H}_k \mathbf{P}_k' \mathbf{H}_k^T &= \mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T & - \mathbf{K}\mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T\end{aligned}$$

相应的，卡尔曼增益就是：

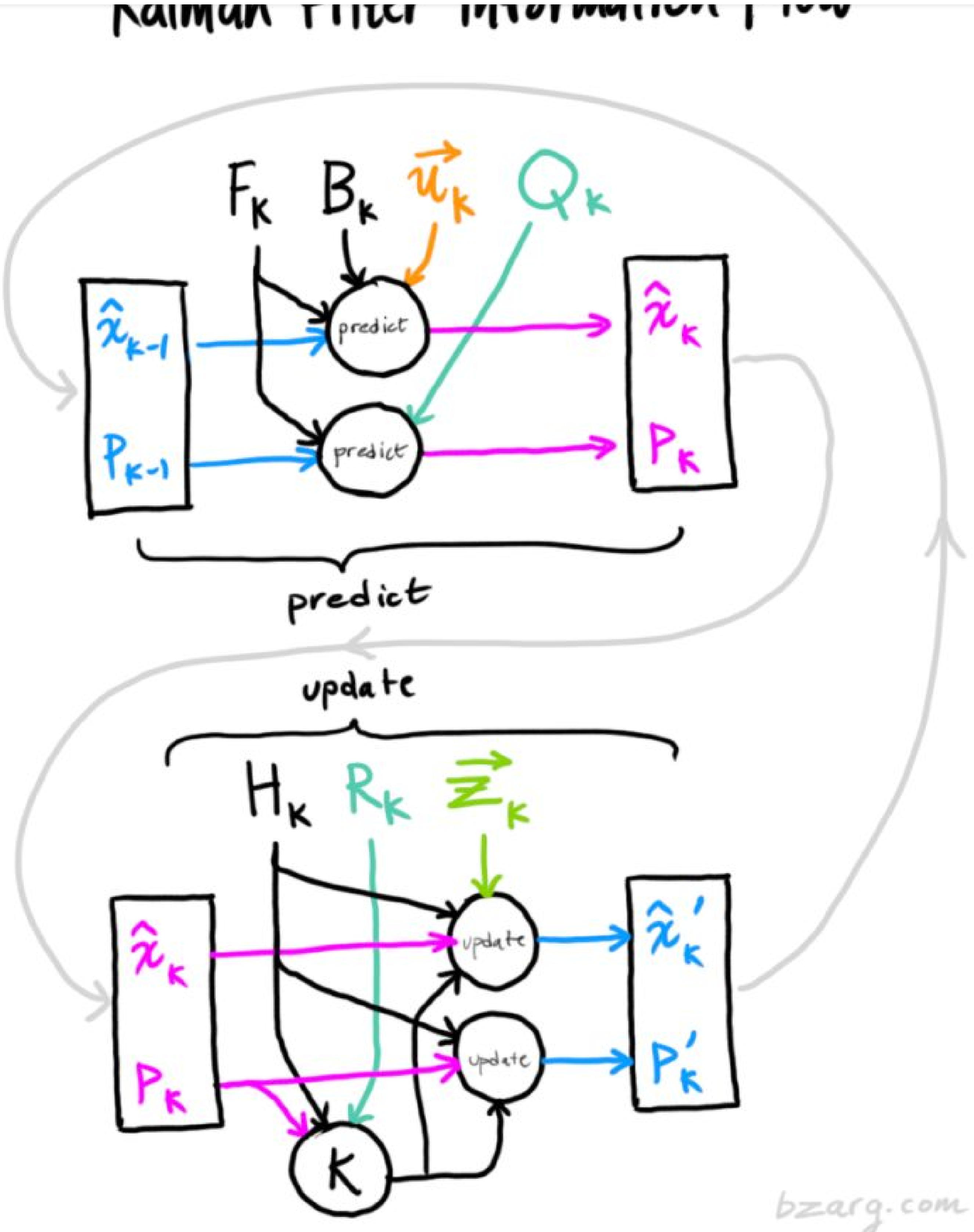
$$\mathbf{K} = \mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

考虑到 K 里还包含着一个 H_k ，我们再精简一下上式：

$$\begin{aligned}\hat{\mathbf{x}}_k' &= \hat{\mathbf{x}}_k + \mathbf{K}'(\vec{\mathbf{z}_k} - \mathbf{H}_k \hat{\mathbf{x}}_k) \\ \mathbf{P}_k' &= \mathbf{P}_k - \mathbf{K}' \mathbf{H}_k \mathbf{P}_k\end{aligned}$$

$$\mathbf{K}' = \mathbf{P}_k \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k \mathbf{H}_k^T + \mathbf{R}_k)^{-1}$$

最后， $\hat{x}_k'$ 是我们的最佳估计值，我们可以把它继续放进去做另一轮预测：



希望这篇文章能对你有用！

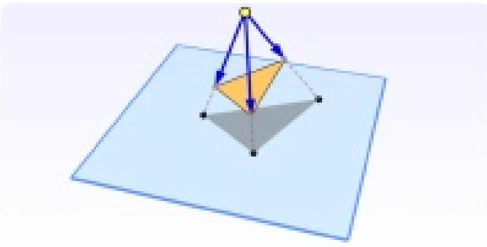
编辑于 2018-07-17

文章被以下专栏收录

论智
专注于人工智能新技术、新应用【公众号：论智（jqr_AI）】

进入专栏

推荐阅读



完结！《沉浸式线性代数》完整版正式发布，全交互式体验

红色石头

傻瓜也能懂的十 译自外网博客）

彦鑫

84 条评论 ⇌ 切换为时间排序

写下你的评论...

superheasy 1 年前
满足矩阵A分布 是什么？概率没学好，没看懂。。。
5

论智 (作者) 回复 superheasy 1 年前
多谢指出，可能当时有点脑抽。。是任意矩阵A，用来解释乘上预测矩阵FK后的变化（小编再去检查下）
8

▲ 赞同 2.2K ▼ 84 条

对协方差矩阵的每一个元素乘一个矩阵，等于这个矩阵乘协方差矩阵再乘这个矩阵的转置矩阵，是这么理解么？都毕业好多年了，而且概率论没怎么学懂，麻烦您能稍微解释下么，感谢！

赞

展开其他 1 条回复

云中客 1 年前

篇篇精品

赞

TY8822 1 年前

能用来炒股吗?

4

ChickFactorydeEd 回复 TY8822 1 年前

不是没有人试过,但是还没见过成功的.首先你得设计炒股过程中运动公式,其次你得设计噪声分布. 这特么谁知道股票走势符合什么物理规律.

6

李sir 回复 TY8822 1 年前

可以，不过会亏死

5

查看全部 10 条回复

he.liu10 1 年前

总感觉这是个预测用的方法，为啥叫滤波器呢

11

LEOO 回复 he.liu10 1 年前

用于多传感器融合中去除噪点，是一种数字滤波方式。类似于图像处理中的滤镜

5

ChickFactorydeEd 回复 he.liu10 1 年前

换个角度,你当做是多传感器的数据融合.视觉,IMU,GPS等等传感器的数据融合过程.

5

查看全部 9 条回复

16

知乎用户 1 年前

66666

赞

知乎用户 1 年前

最近在学习机器人，文章非常不错，一维的高斯合成看了之后很有启示，建议说明一下卡尔曼滤波的假设条件呢

3

ChickFactorydeEd 1 年前

所以 啥时候讲数据融合和各种偏导?

2

神隐博士 1 年前

这篇文章我见过不下10次了。。。

3

甄景贤 1 年前

covariance 是什么、Cov(x) 是什么、解释得不够，我卡住了，要上维基百科查，……迟些再看。

赞

沙小沙 1 年前

第一次看到这个文章。第一次从公式理解角度看懂了卡尔曼滤波

1

知乎用户 1 年前

最近看ekf slam看的头痛

赞

知乎用户 1 年前

卡尔曼增益讲的不错

赞

知乎用户

你有被授权吗

苏格拉顶 1 年前

大写的Pk是什么？

赞

SLee 回复 苏格拉顶 1 年前

协方差矩阵

赞

知乎用户 1 年前

所以 这预测和更新步骤 和 slam中基于非线性优化方法 设置目标函数的思路有异曲同工之妙啊

2

MechCD 1 年前

要是有个案例和相应的代码就好了

赞

Eric.Zhou 1 年前

后面有点不太清晰。

1

聊招晨梦为鹤 1 年前

请问过程噪声呢，它应该在协方差矩阵P的更新过程中体现的吧

赞

知乎用户 1 年前

我应该看过英文版，这是翻译的？

4