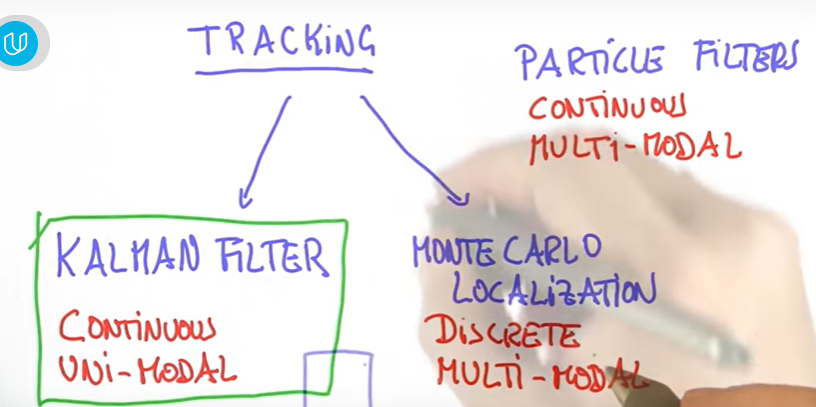
**Kalman Filters**

Link video 1: <https://www.youtube.com/watch?v=2zmbIjHpkRM>

1. **Tracking Intro:**



Video: <https://www.youtube.com/watch?v=BkjQzEyJWrE>

Probability density function tutorial: <https://www.khanacademy.org/math/statistics-probability/random-variables-stats-library/random-variables-continuous/v/probability-density-functions>

**Kalman Filter** = technique for estimating the state of a system.

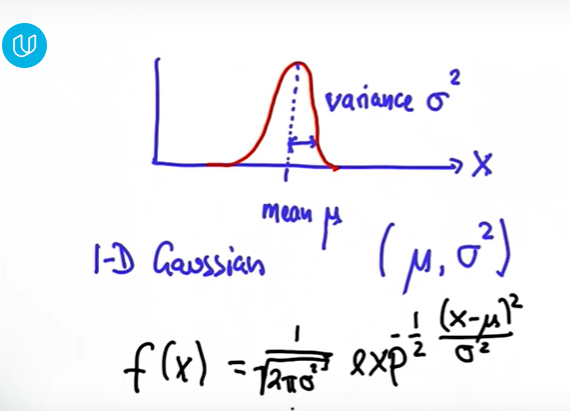
Yb video: <https://www.youtube.com/watch?v=mwn8xhgNpFY&t=13s>

Morala: Sa zicem ca te plimbi cu masina si folosesti ca senzori GPS-ul etc. Daca Masina intra intr-un tunel, teoretic nu o vei mai putea localiza. Folosind Kalman filter, vei putea sa estimezi pozitia masinii in tunel in functie de datele furnizate de senzori ultima data cand o detecta.

Cuvinte:

1. Uni modal distribution
2. **Gaussian Intro:**

Video: <https://www.youtube.com/watch?v=6IhtnM1e0IY>



In Kalman Filter, distributia este gausiana (functie continua, iar spatiul de sub aceasta are suma = 1).

El este caracterizat de:

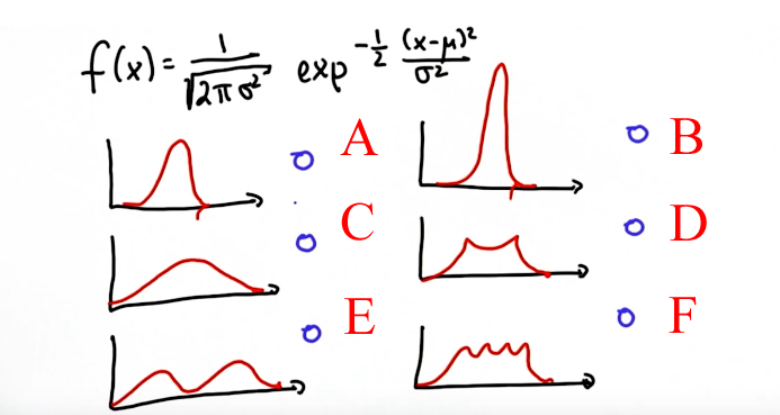
1. de un mean
2. de un width of the Gaussian (numita si variance) notat cu sigma la patrat

! any Gaussian is 1D

Taskul nostrum in Kalman filter este sa mentinem un (mu, Sigma^2) ca cea mai buna estimare a locatiei pe care dorim sa o gasim.

Aceeasi formula este la fel ca exponentiala, vezi figura de mai sus. Iar constata ce se inmulteste cu exponentiala, este doar o simpla constanta cu scopul de a normaliza functia.

Exercitiu:



* doar A,B,C sunt functii Gausiene

1. **Quiz: Variance Comparation**

Video: <https://www.youtube.com/watch?v=TGdMG81hXc8>

1. **Quiz: Preferred Gaussian**

Video: <https://www.youtube.com/watch?v=sBsju-6nQWI>

Mora Quizului: Daca mergi cu masina, cu cat “Variance” este mai mic, cu atat ce dorim noi sa aflam este mai precis.

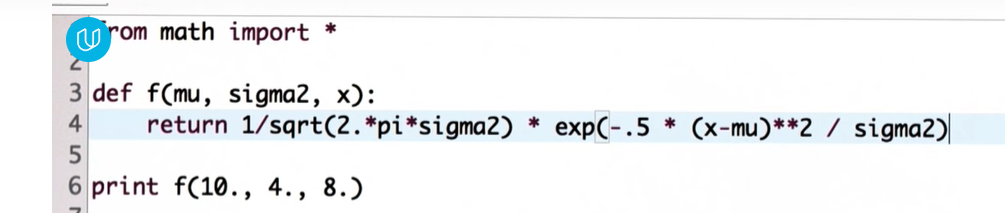
1. **Evaluate Gaussian**

Video: <https://www.youtube.com/watch?v=4-0nBfsD4jo>

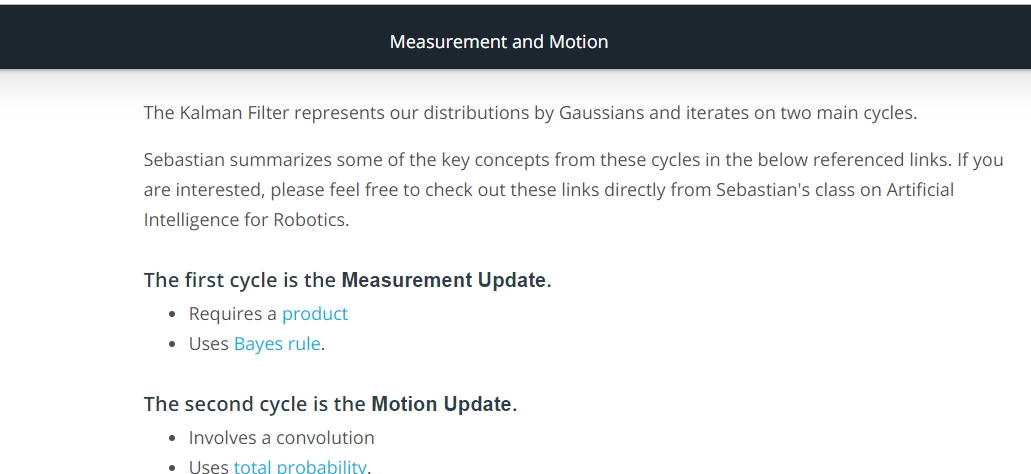
Video: <https://www.youtube.com/watch?v=mQtjczyAxQs>

1. **Maximize Gaussian**

Video: <https://www.youtube.com/watch?v=fRYtUP0P4Lg>

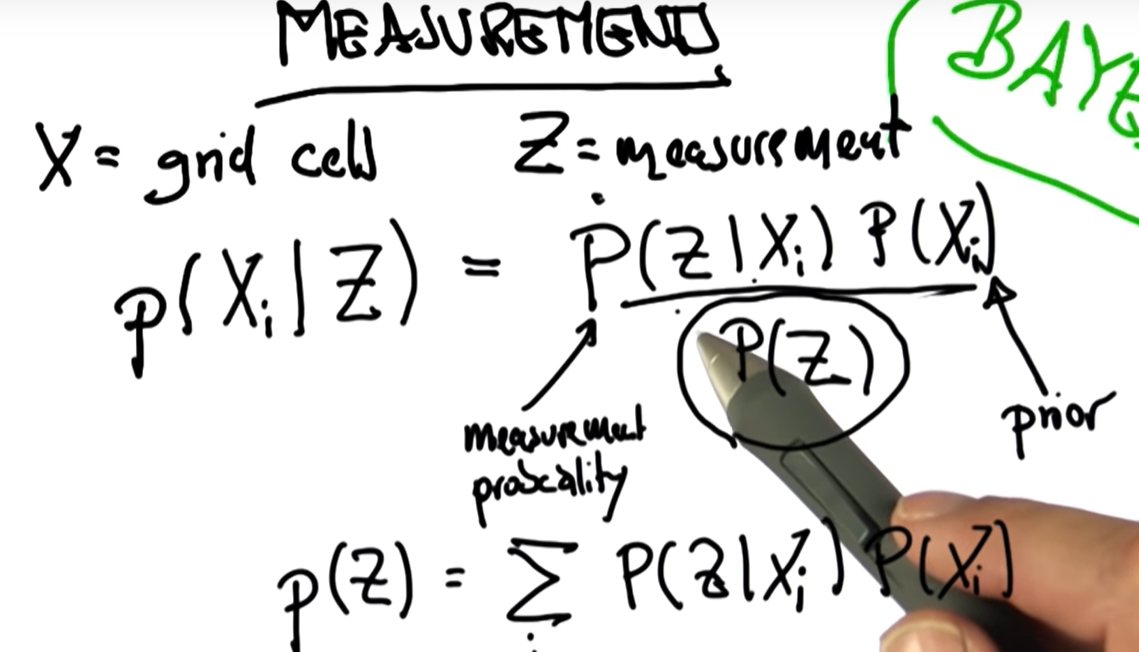


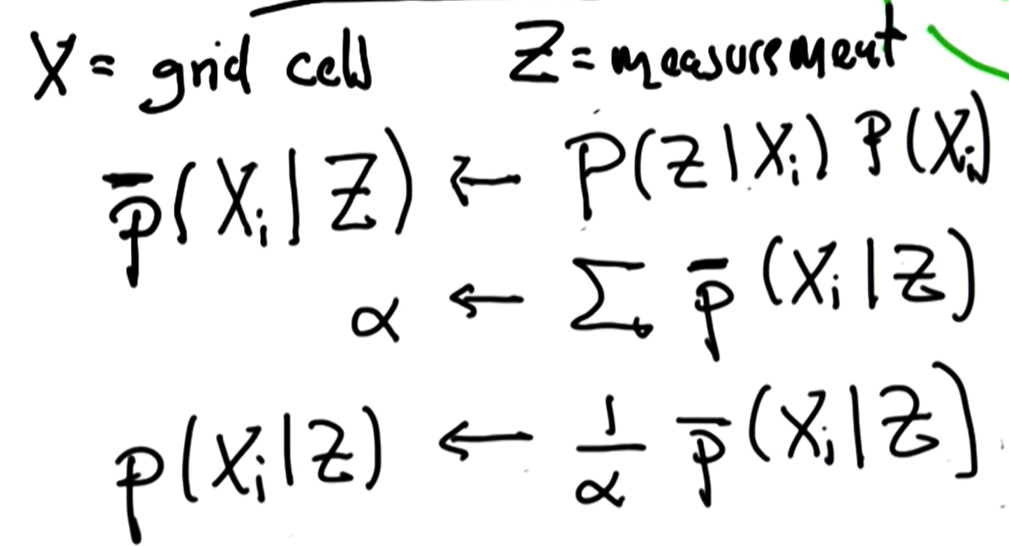
1. **Measurement and Motion**



Link 1: <https://www.youtube.com/watch?v=UFcTLCttNRI>

Link2(Bayes Rule): <https://www.youtube.com/watch?v=sA5wv56qYc0>





Link(Total Probability): <https://www.youtube.com/watch?v=n1EacrqyCs8>

P(z) = este practic probabilitatea de a vedea o masuratoare, in lipsa oricarei informatii cu privire la localizare.

P(z) = doesn’t have the grid cell as index.

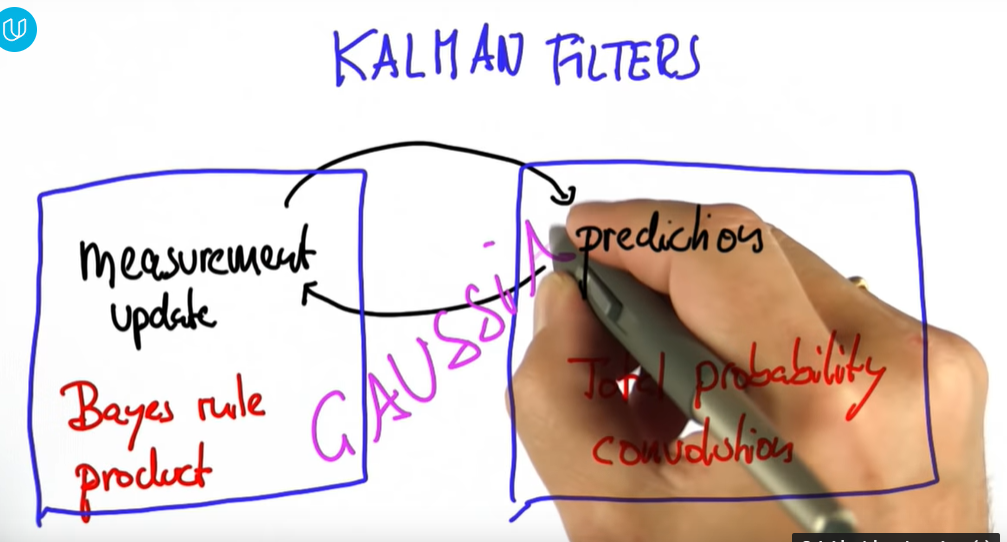
Daca normlizam, putem sa calculam exact p(z)

Alpha = the normalizer

P cu bara = un non normalize probability, si asupra lui pune produsul ala ce il vezi in p(z)

1. **Quiz Shifting the Mean**

Video: <https://www.youtube.com/watch?v=8c479K2UCZo>



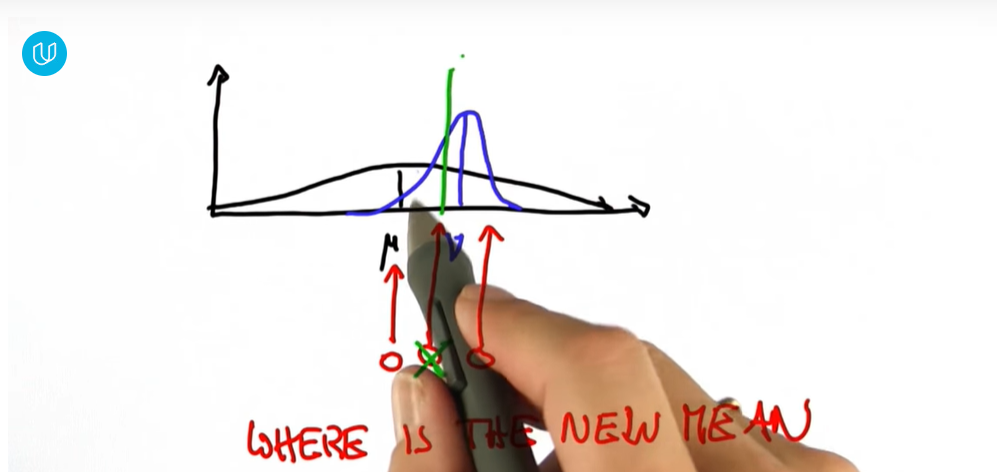
Kalman Filters iterate: “measurement and motion”

Measurement = often called measurement update

Motion = often called prediction

Bayes rule = it’s a product of multiplication

For Prediction, we use “total Probability” witch is a “convolution” or simply an addition.



EX:

Suppose you’re localizing another vehicle, and you have a distribution like in the image.

Si sa zicem ca mai ai o masuratoare dea ta(ce e cu albastru) ce ti-ar da localizarea vehiculului.

1. **Quiz: Prediction the Peak**

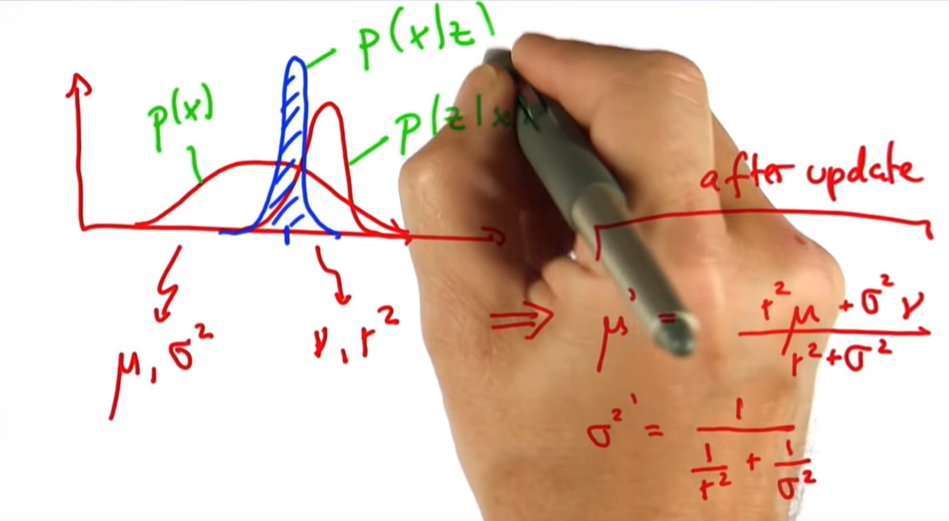
Link Video: <https://www.youtube.com/watch?v=zc_GQiISQ3E>

1. **Quiz: Predicting Peak**

Video: <https://www.youtube.com/watch?v=zc_GQiISQ3E>

1. **Parameter Update:**

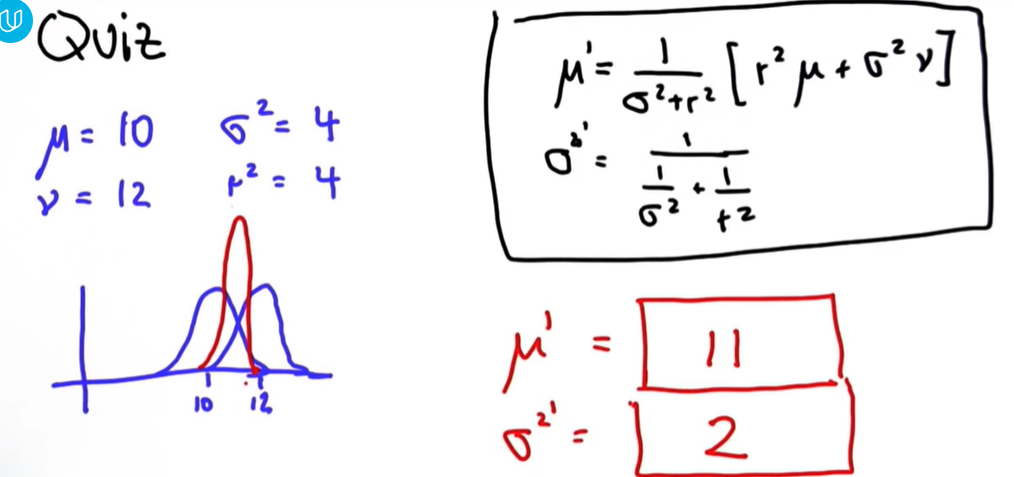
Video: <https://www.youtube.com/watch?v=d8UrbKKlGxI>



**P(x)** -> prior

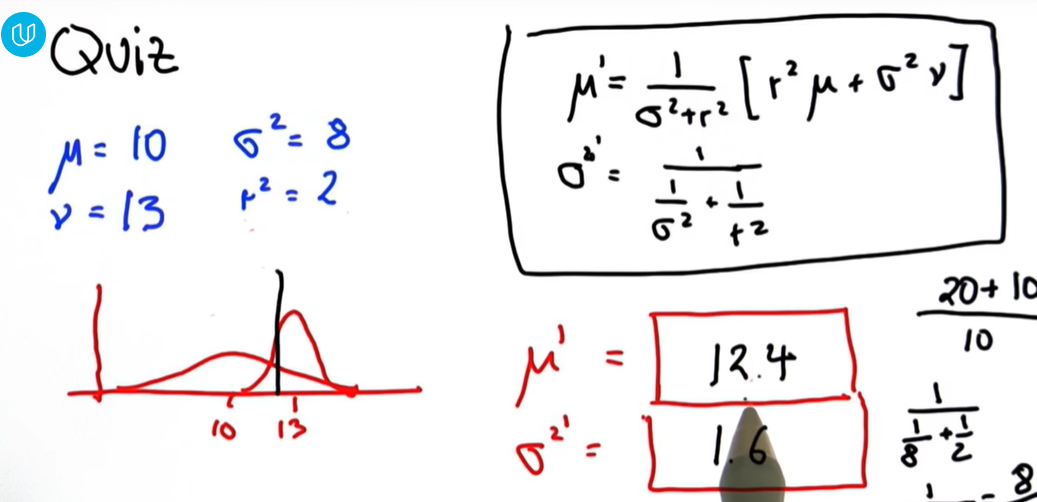
**P(z|x)** -> measurement probability

**P(x|z)** -> posterior



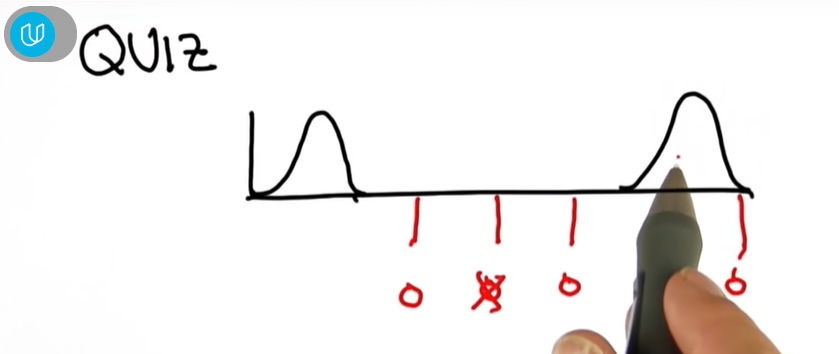
1. **Quiz Parameter Update 2**

**Video:** [**https://www.youtube.com/watch?v=2BfisMbu86o**](https://www.youtube.com/watch?v=2BfisMbu86o)



1. **Quiz Separated Gaussians:**

**Video:** [**https://www.youtube.com/watch?v=QAqsIWVVX0Y**](https://www.youtube.com/watch?v=QAqsIWVVX0Y)



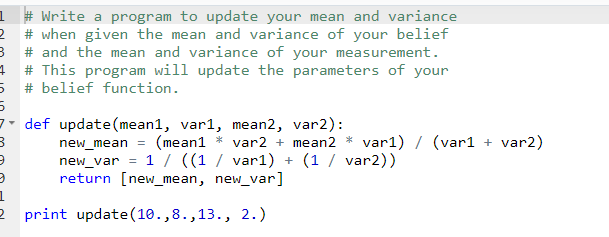
**Unde este midul?**

1. **Quiz: Separated Gaussians 2**

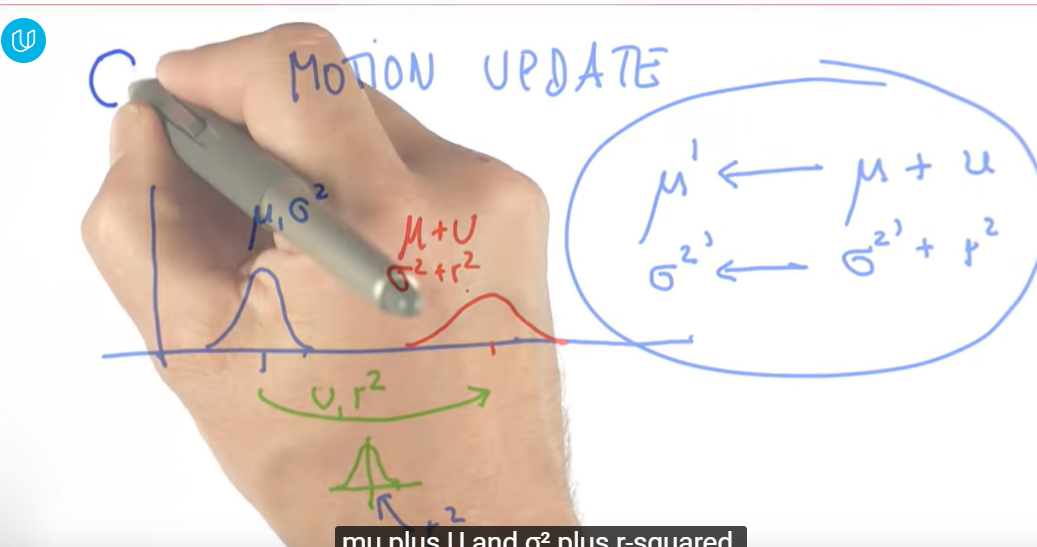
**Video:** [**https://www.youtube.com/watch?v=0FmTokjoRgo**](https://www.youtube.com/watch?v=0FmTokjoRgo)

1. **Quiz: New Mean and Variance:**

**Video:** [**https://www.youtube.com/watch?v=yo8jf0U4hlc**](https://www.youtube.com/watch?v=yo8jf0U4hlc)



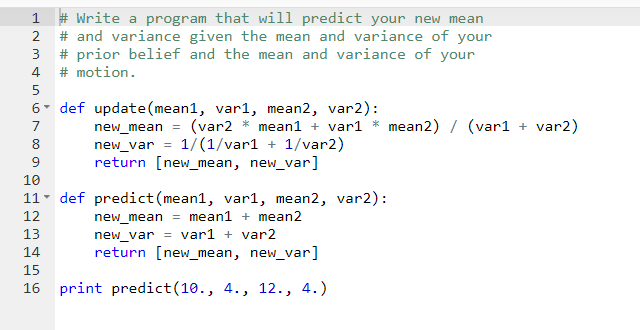
1. **Quiz: Gaussian Motion**



**Video:** [**https://www.youtube.com/watch?v=X7YggdDnLaw**](https://www.youtube.com/watch?v=X7YggdDnLaw)

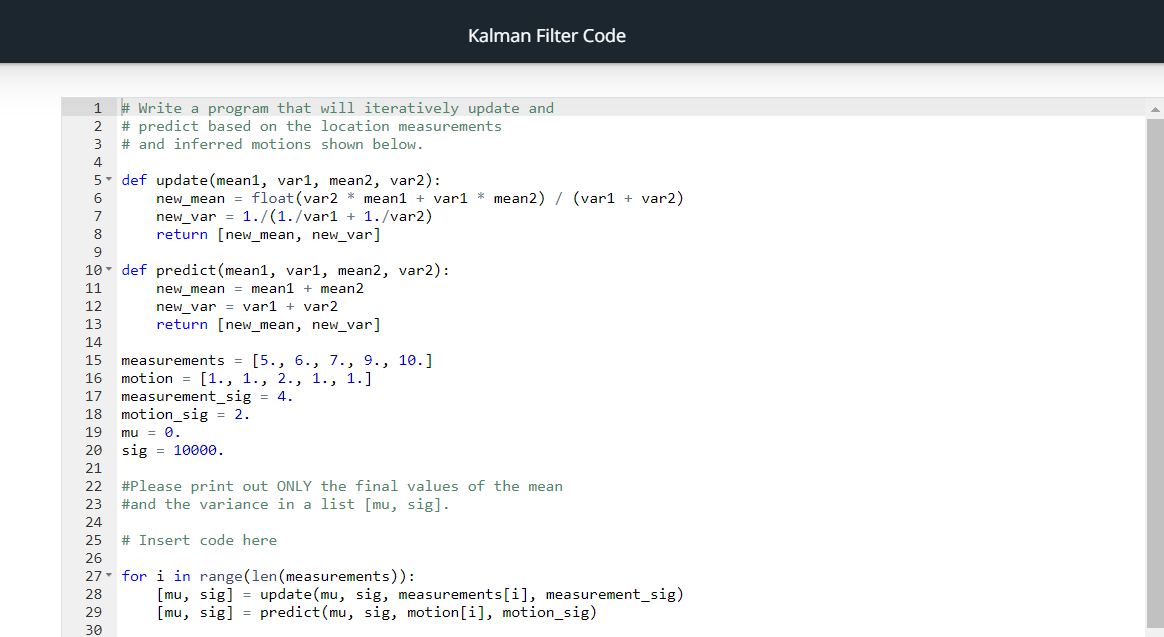
1. **Quiz: Predict Function**

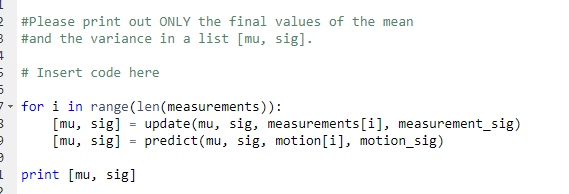
**Video:** [**https://www.youtube.com/watch?v=DV2cX9W0tT8**](https://www.youtube.com/watch?v=DV2cX9W0tT8)



1. **Quiz: Kalman Filter Code**

**Video:** [**https://www.youtube.com/watch?v=3xBycKfnCOQ**](https://www.youtube.com/watch?v=3xBycKfnCOQ)





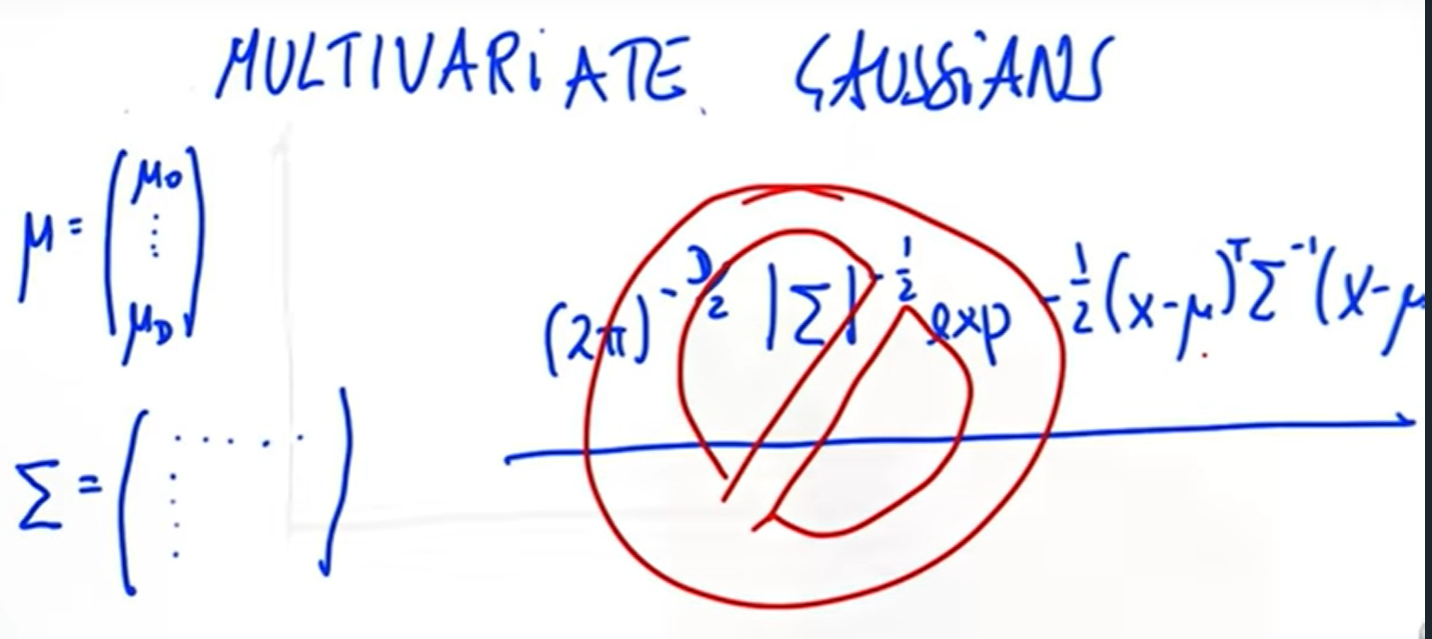
1. **Quiz: Kalman Prediction**

**Video:** [**https://www.youtube.com/watch?v=doyrdLJ6rJ4**](https://www.youtube.com/watch?v=doyrdLJ6rJ4)

1. **Kalman Filter Land**

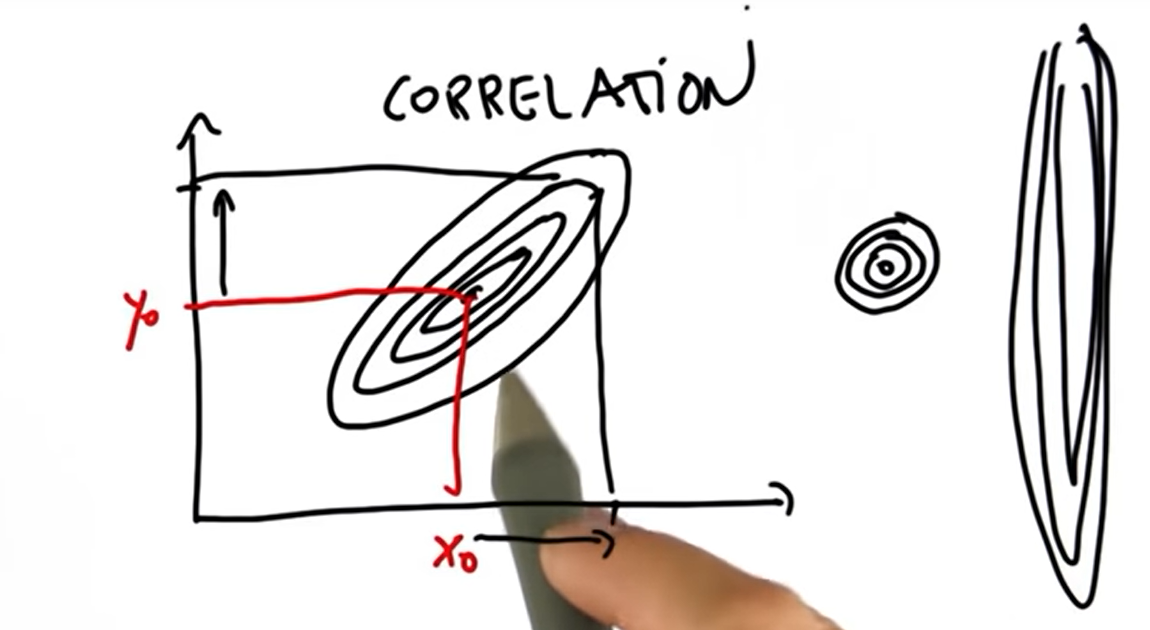
**Video:** [**https://www.youtube.com/watch?v=LXJ5jrvDuEk**](https://www.youtube.com/watch?v=LXJ5jrvDuEk)

**High dimensional Gaussians** = se mai numesc si **MultiVariate Gaussians**



**Mean** = este un vector acum

**Variance** = este inlocuita de o co-varianta ce este o matrice cu D randuri si D coloane. Daca dimensiunea estimate este D



1. **Quiz: Kalman Filter Prediction:**

Video: <https://www.youtube.com/watch?v=HTL5-0DDqE4>

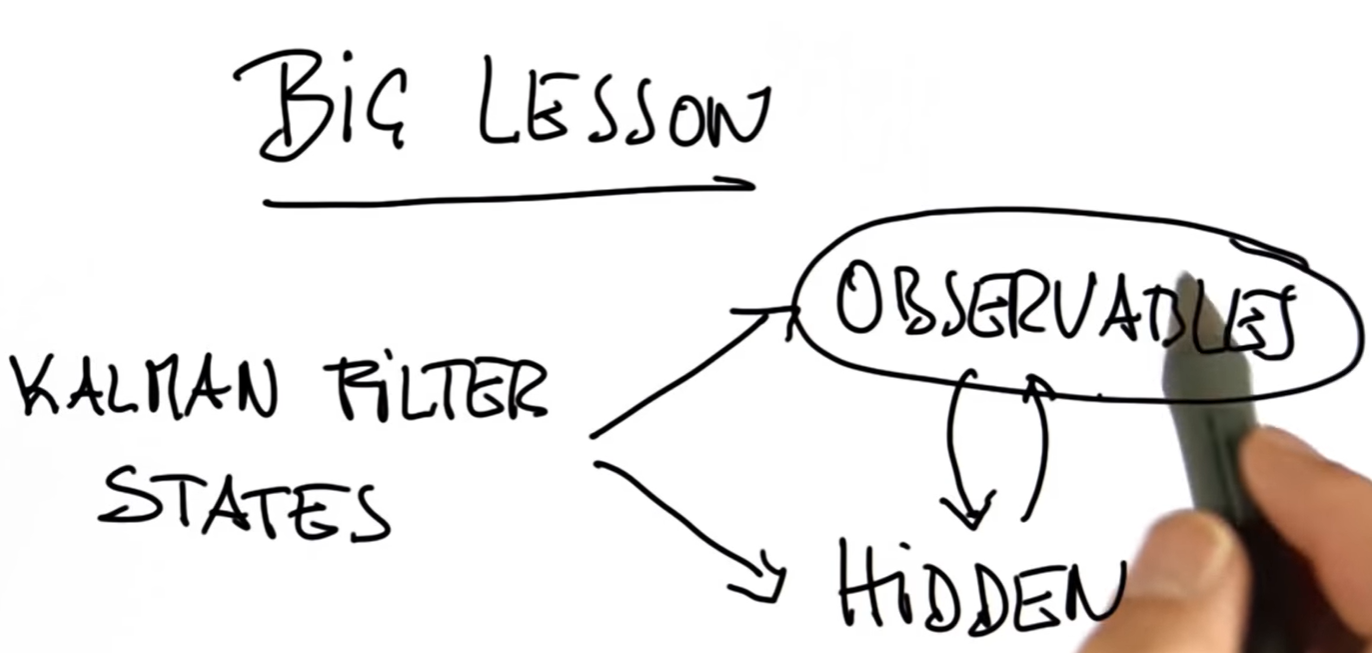
Rezolvare: <https://www.youtube.com/watch?v=SK3cnmu8BYU>

1. **Quiz: Another Prediction:**

Video: <https://www.youtube.com/watch?v=cUKlYjQEQGY>

1. **More Kalman Filters:**

Video: <https://www.youtube.com/watch?v=hUnTg5v4tDU>



1. **Kalman Filter Design**

Yb: <https://www.youtube.com/watch?v=KYEr4BXhD_E>

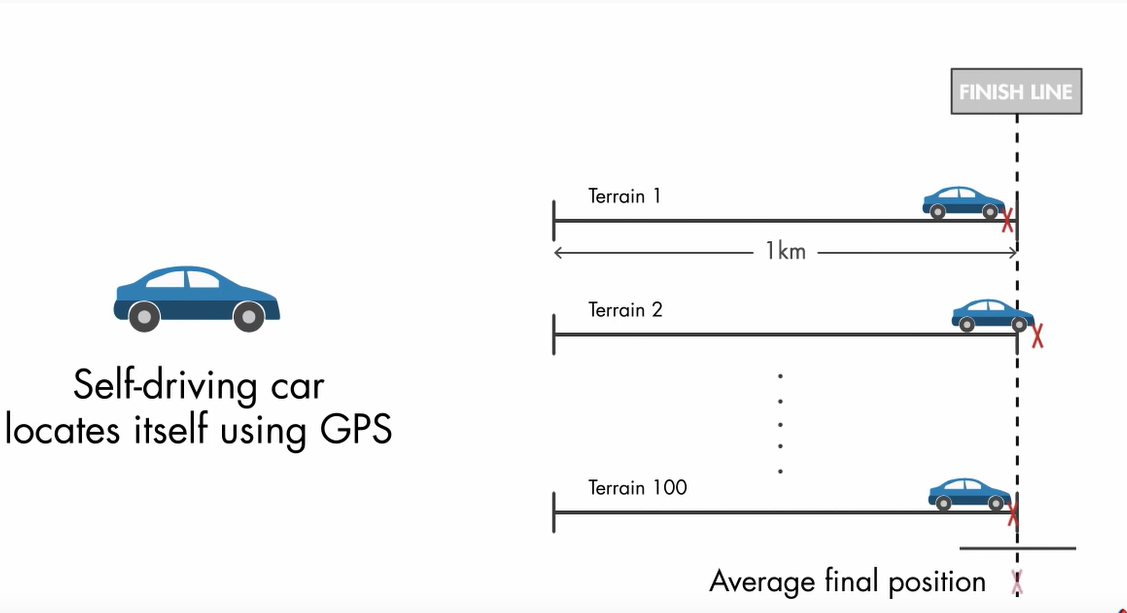
1. Kalman Filter Matrices:

Yb: <https://www.youtube.com/watch?v=ade97UKqSIc>

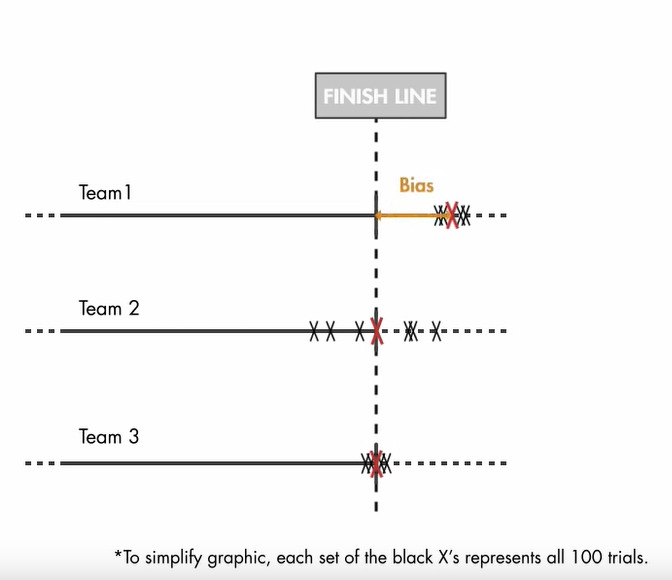
Kalman filder Calculations: <http://www.ilectureonline.com/lectures/subject/SPECIAL%20TOPICS/26/190>

**Eu Kalman Filter**

Video playlist cu tutoriale = <https://www.youtube.com/watch?v=mwn8xhgNpFY&list=PLn8PRpmsu08pzi6EMiYnR-076Mh-q3tWr>



* Sa zicem ca este la o competitie si premiul se acorda aceluia care se opreste perfect, cat mai aproapte de linia de finish.
* La final, cel cu varianta cel mai mica, fata de medie CASTIGAAA!!



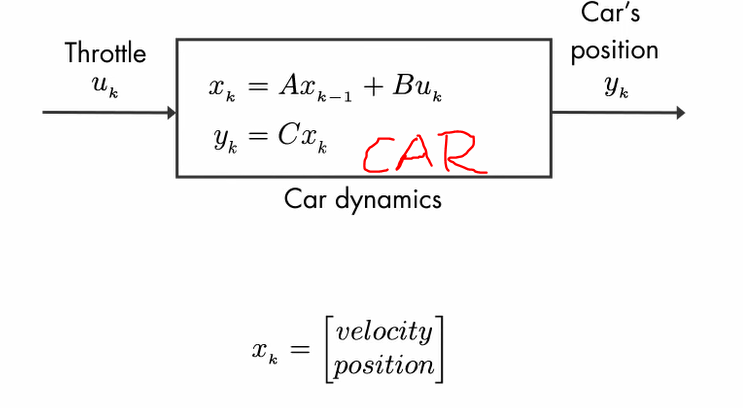
* Sa spunem ca avem mai multe echipe(1,2,3). Fiecare jucator din fiecare echipa, s-a oprit in locul marcat cu “X”, iar media este notata cu un X rosu.
* Bias = distanta dintre Finish Line si media
* Variance = este cat de aproape sunt finish line X(negre) fata de medie X(rosu)



* Se vede clar ca pentru Team 1, avem o varianta mica, in schimb bias-ul este foarte mare.
* Team 2: are bias mic, dar varianta mare
* Team3: castiga, pt ca are o varianta mica, iar media e fix pe centru

Daca vrei sa castigi, trb sa te bazesi pe mai multi senzori, nu doar pe GPS. Ca sa indeplinesti criteriile competitive, vei folosi un Kalman filter pentru a estima pozitia masinii.

Hai sa intelegem cum functioneaza:



Input-ul masinii = Throttle (ro: regulator)

Outputul: yk = pozitia masinii.

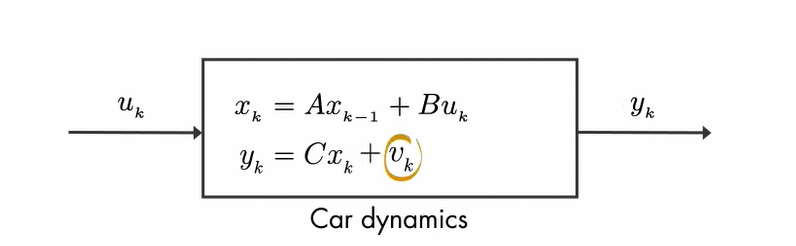
Pentru un astfel de sistem vom avea multe “stari” Xk = [velocity, position], dar pentru a simplifica lucrurile pt un astfel de sistem, vom presupune:

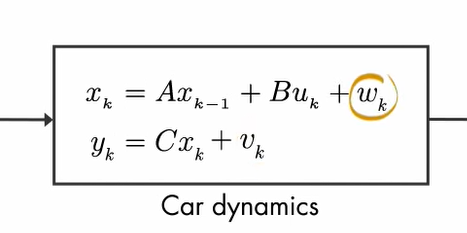
Inputul masinii = este velocity (viteza). In acest fel vom avea un singur state, si anume xk = [position] si vom masura starea, deci matricea C = 1.

Ce dormim noi? R: ca masina sa fie foarte precisa si sa se opreasca cat mai aproape de Finish line.

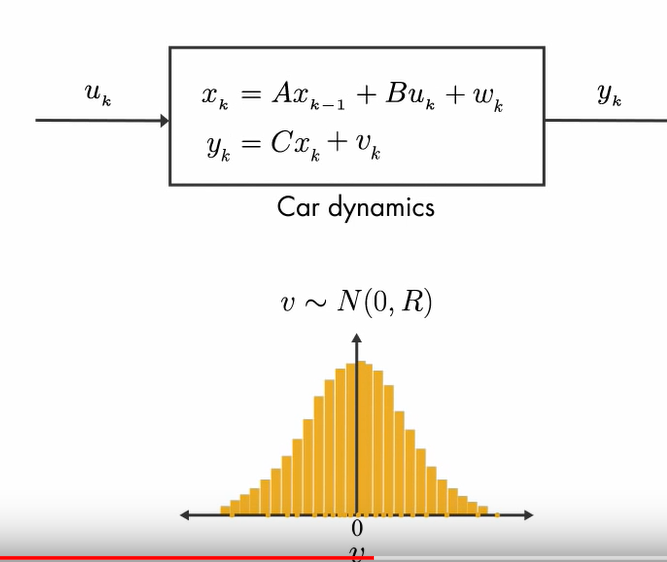
Dar, GPS signal va fii noisy.

Fie vk = masura acestui zgomot al GPS-ului (fiind o variabila random)





De asemenea putem avea si un Wk = care ar fii zgomotul produs de vant si ar schimba viteza masinii. Il vom presupune random si pe asta.

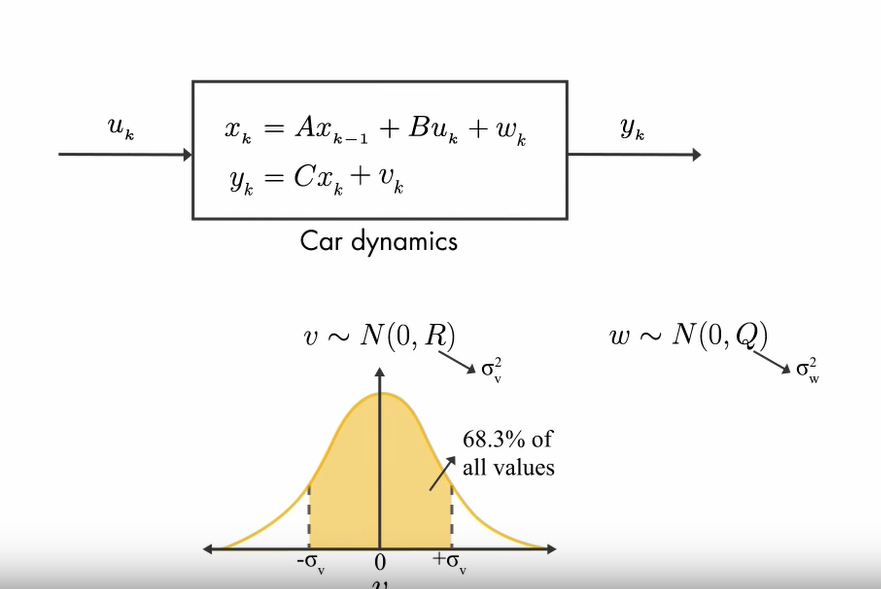


Desi aceste variabile, nu urmeaza un anumit pattern, ne putem folosi de teoria probabilitatilor, pentru a estima media acestor proprietati

v = de exemplu, este extras dintr-o distributie Gaussiana cu mean = zero si covarianta R.

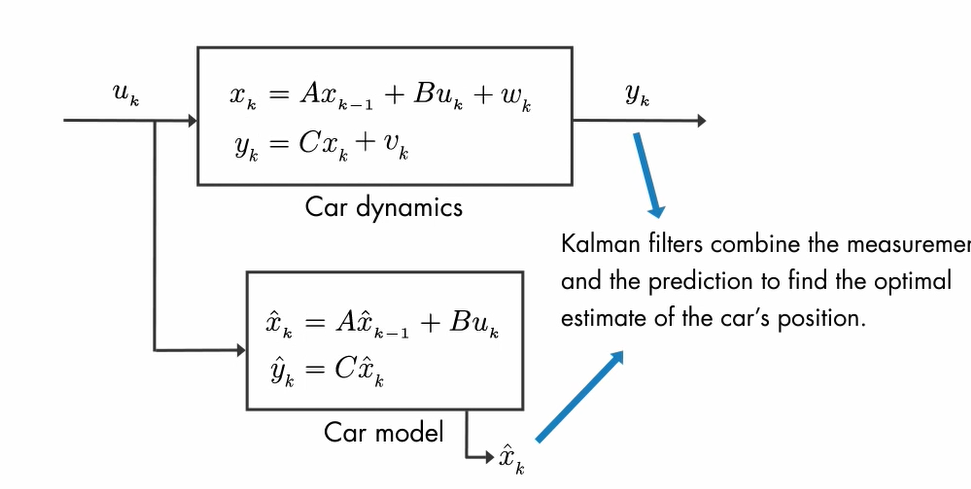
De exemplu, daca masuram pozitia masinii sa spunem de 100 de ori in aceeasi locatie, zgomotul din aceste citiri va lua valorile cele mai apropiate de mean = zero, si mai putine pe cele indepartate.

Asta rezulta intr-o distributie Gaussiana, ce este reprezentata de covarianta R.



Din moment ce avem un sistem cu un singur output, Covarianta ‘R’ este scalare si egala cu varianta “of the measurement noise”.

Similar, procesul de zgomot, este de asemenea random si realizeaza o distributie Gausiana cu covarianta **Q**.



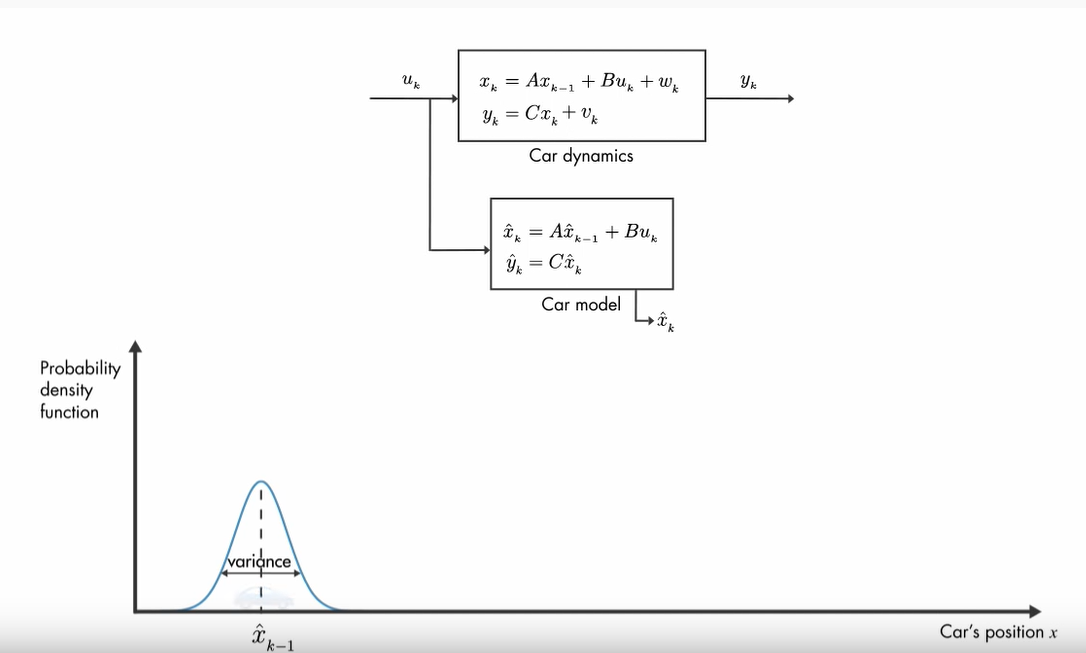
Daca stim modelul masinii, putem introduce inputul in el, astfel incat sa ii estimam pozitia. Ce e acolo cu caciulita inseaman estimarea.

Dar nici aceasta metoda nu va fii foarte perfecta, asta deoarece, noi estima un x cu caciula, ce este nesigur datorita zgomotului ambiental.

Aici intervine Kalman Filter: ce va combina aceste doua bucati de informatie si va venii cu o mai buna estimare a masinii.

Vom vizualiza acest principiu a lui Kalman filter visual, cu ajutorul functiilor “probability density functions”.

**Deci,**

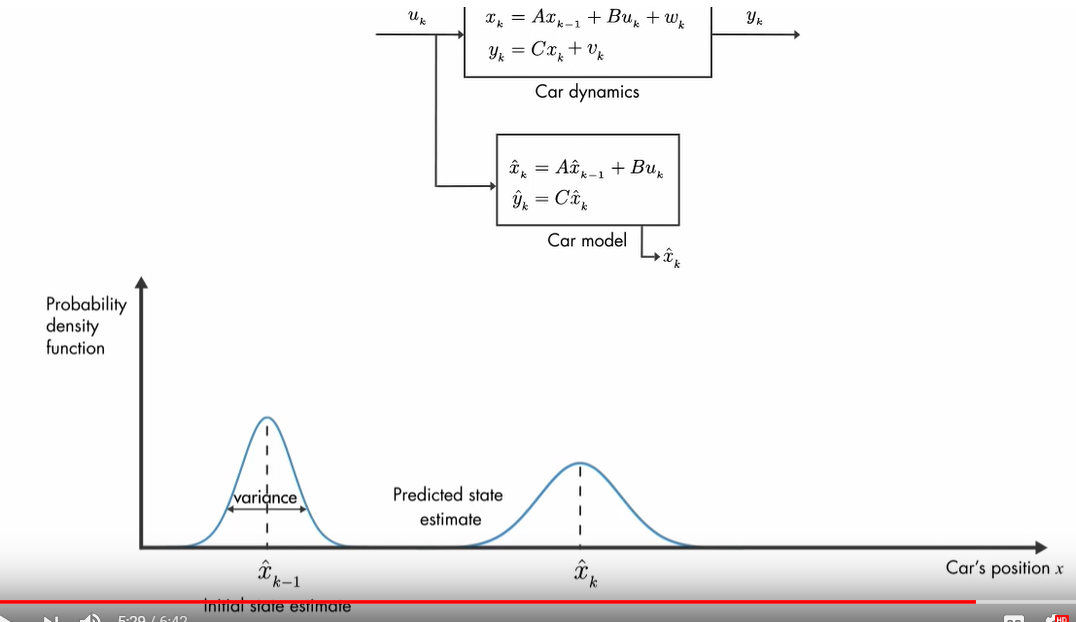


**Initial time step** = k-1 (pozitia masinii poate sa fie oriunde, in jurul estimarii x^hat\_(k-1)

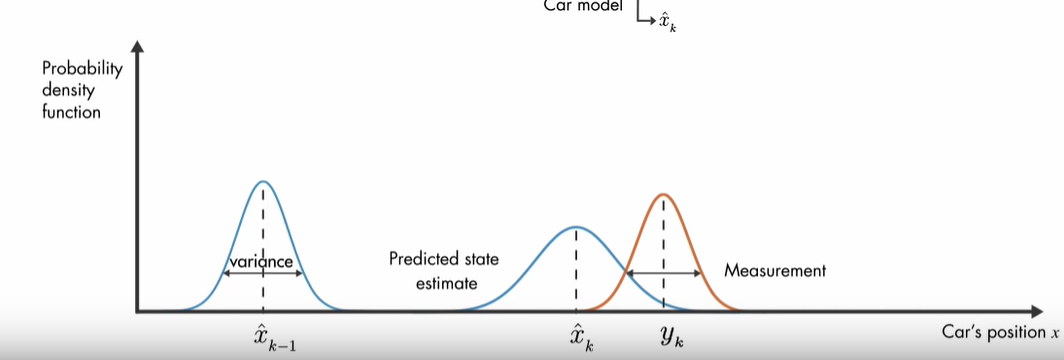
Acest **“uncertainty”** este descries de: **Probability density function**

Ce ne mai spune acest plot, este ca, masina ar fii cel mai probabil in jurul mediei (mean) distributiei.

**La pasul urmator,** incertitudinea in estimare a crescut (a se observa varianta larga). Asta deoarece, intre pasul **k-1** si **k**, masina poate a dat intr-o groapa, sau s-a intamplat ceva cu rotile. Fapt pentru care, a parcurs o distanta diferita de cea pe care a preziso modelul.



Cum am discutat si inainte, o alta sursa de informatie a pozitiei masinii vine dintr-un alt measurement.



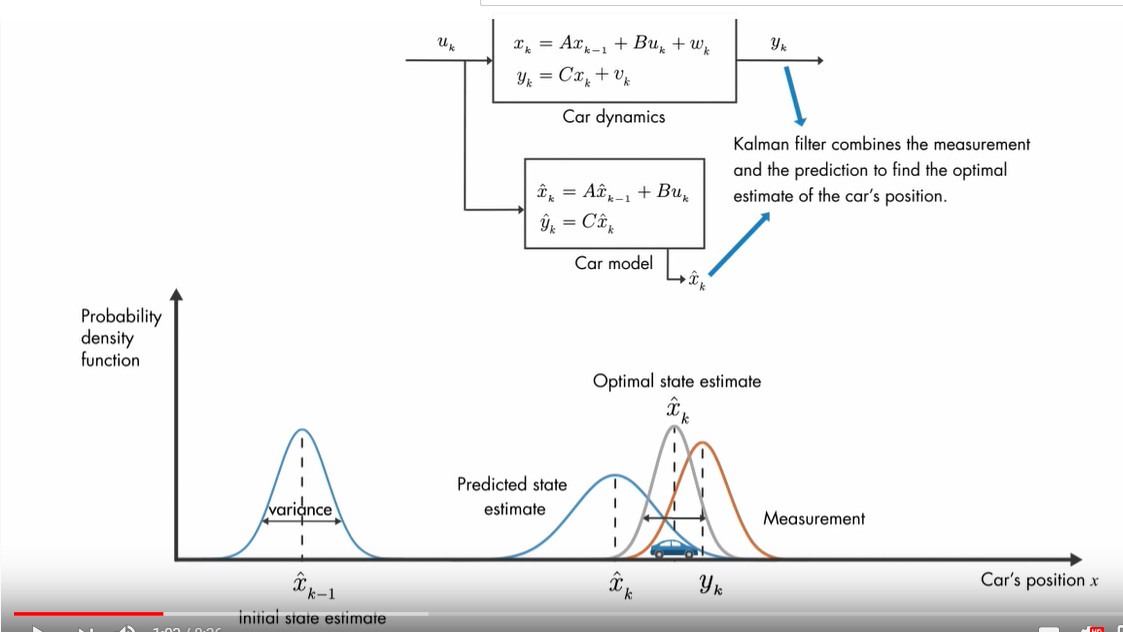
Aici, **varianta** = reprezinta, incertitudinea in masurarea noizului. (adik linia aia de la measurement)

Din nou, adevarata pozitie poate fii oriunde in jurul mediei.

Deci ca sa ne intelegem: ai pornit dintr-o pozitie initiala, dar si acea pozitie initiala cand o calculezi zgomotul din jur o poate influenta, de asta ai tu acea distributie albastra in stanga, unde varianta reprezinta incertitudinea asta a masurarii. Apoi tu pornesti masina si vrei sa prezici unde ar ajunge (intre timp ea poate da intr-o groapa etc.), asa ca x\_hat(k) are varianta mai mare, ca in prezicerea ta poate aparea orice. De asemenea, pe baza senzorilor tai, tu masori ca ai ajunge la pozitia yk (desi acestia pot fii si ei influentati de zgomot, ai o distributie si aici). Intr-un final ai 2 distributii, x\_hat(k) (pt masurarea prezisa de la senzori) si y(k) (pt masurarea efectiva)

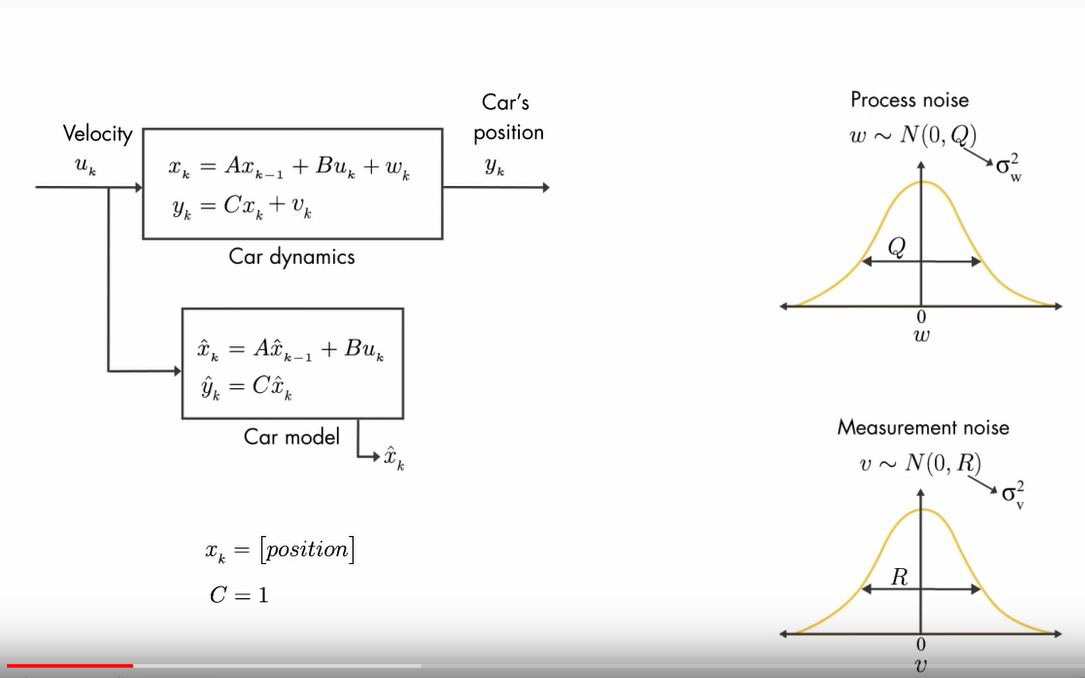
Acum ca avem: **prediction** si **measurement. Intrebarea este, care este cea mai buna estimare a pozitiei masinii?**

Se pare ca, cea mai optima estimare a pozitiei masinii este sa combinam aceste 2 informatii. Iar acest lucru se realizeaza prin multiplicarea acestor doua functii de probabilistica => **Avand ca resultat tot o functie Gaussiana.**



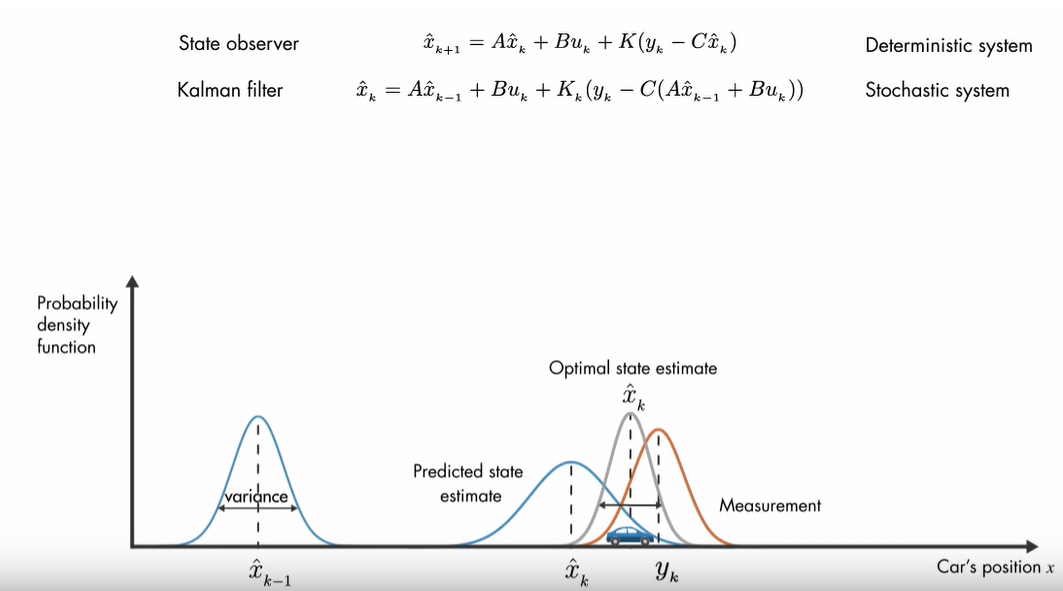
* Aceasta estimare, va avea o varianta mai mica fata de starile anterioare. Iar, media acestei functii probabilistice, ne da estimarea optima pentru pozitia masinii.
* Deci vom inmultii cele doua, scalam rezultatul si calculam media rezultata in urma “probability density function”.

**Optimal State Estimator Algorithm**

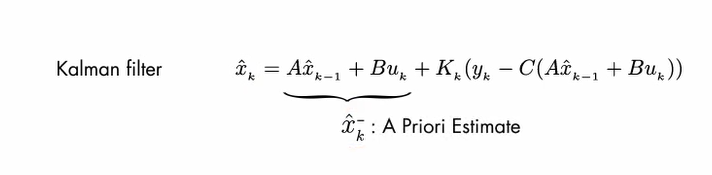


Anterior, am aratat o masina si un model de masina, pentru un sistem cu o singura stare (si anume pozitia) si am discutat procesul si masurare zgomotelor in functie de covarianta.

Computational, aceasta inmultire a “probability density functions” in functie de ecuatia “discrete Kalman filter equation” din imagine, ce seamana cu ecutatia “State observer”.

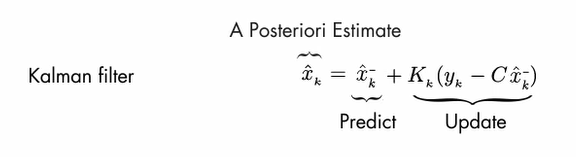


Actually, a Kalman filter is a type of state observer, but it is designed for stochastic systems.



Prima parte x(k), prezice starea curenta folosind estimarea starii din pasul anterior si inputul current.

“x(k) =” si “x(k) numit si **Prior Estimate**” pus acolo cu acolada, sunt diferite ! si il va nota cu x(k) dar cu o bara sus.



* Deci vom putea reduce functia ca mai sus.

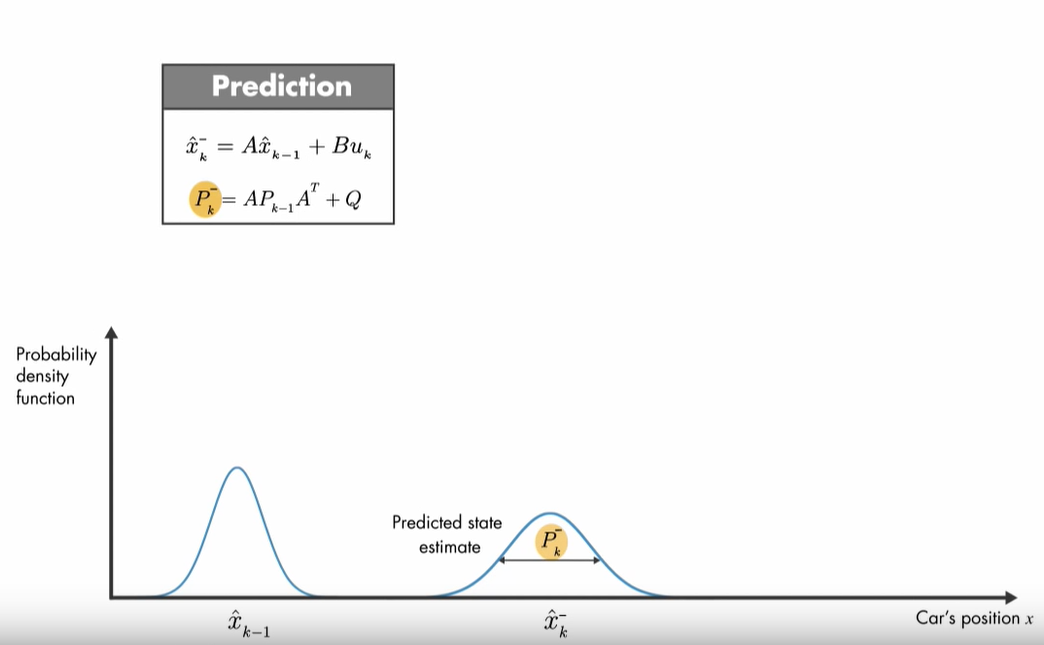
Urmatorul pas este sa luam si “measurement” y(k) ce o bagam in prediction, pentru a updata un “prior estimate” si vom numi rezultatul ca “A Posteriori Estimate”.



* Deci cam alea de sus sunt formulele de care ai nevoie

Algoritm (are 2 steps):

Phase-1:

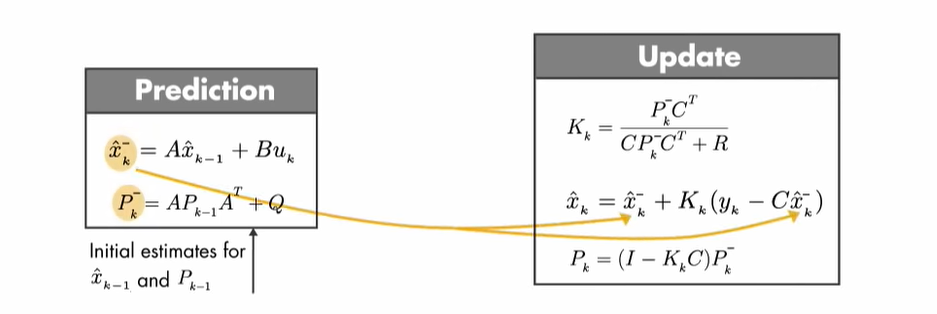


Here, the system model is usded to calculate the prior state estimate and the error covariance P.

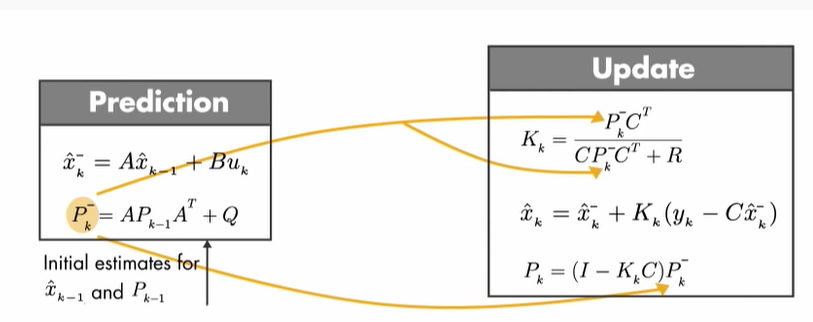
For a single state system, P = is the variance of the prior estimate. And, it can be thought of as a measure of uncertainty in the estimated state.

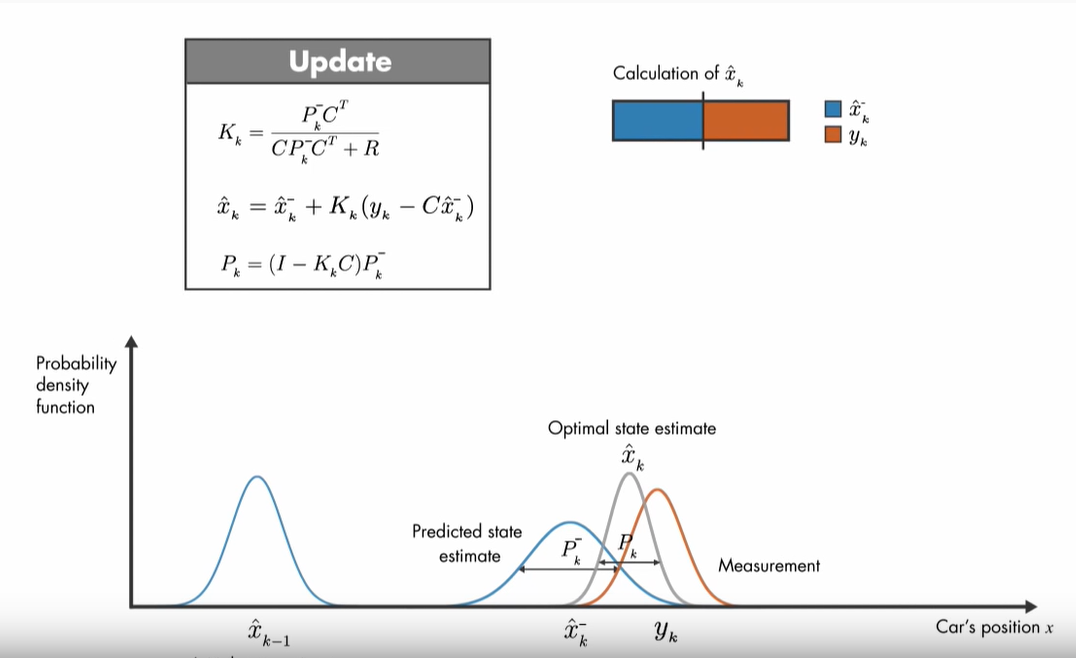
This variance, comes from the process noise, and propagation of the uncertaint x^hat\_(k-1)

At the very start of the algorithm, the k-1 values, for x^hat and P, come from their initial estimates.



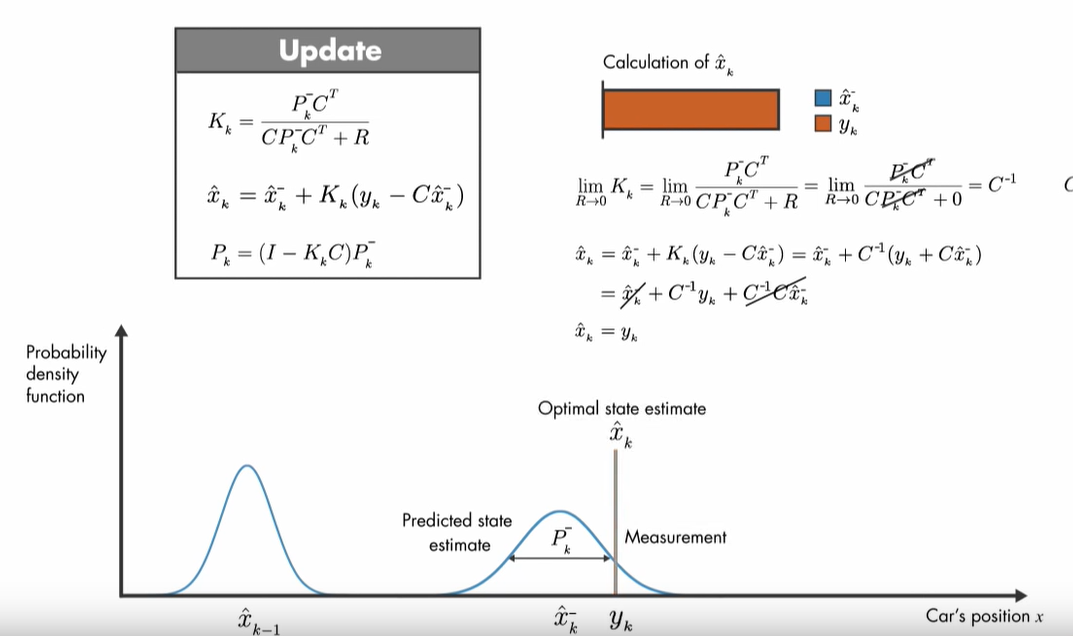
The second step of the algorithm, uses the priori estimates calculated in the prediction step and updates them. To find the posteriori estimates of the state and error covariance.





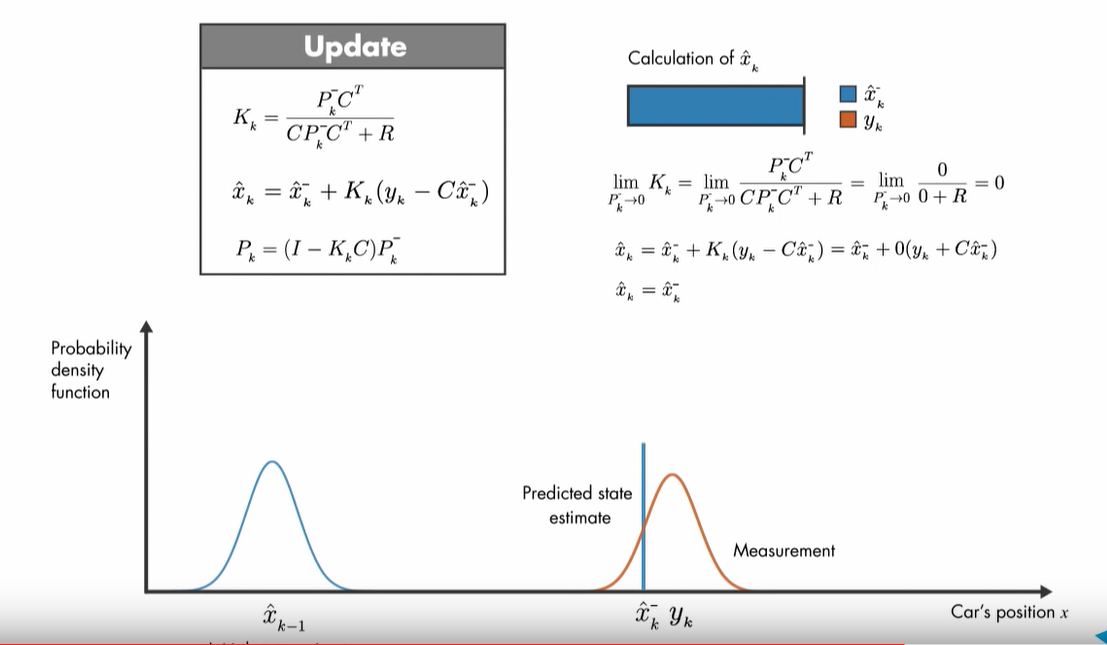
The Kalman gain, is calculated such that if minimizes the posteriori error covariance,

Acolo in bara aia zice, ca se muta de la stanga la dreapta in functie cum este eroarea la masurarea “measurement” sau x(k) cu linie.

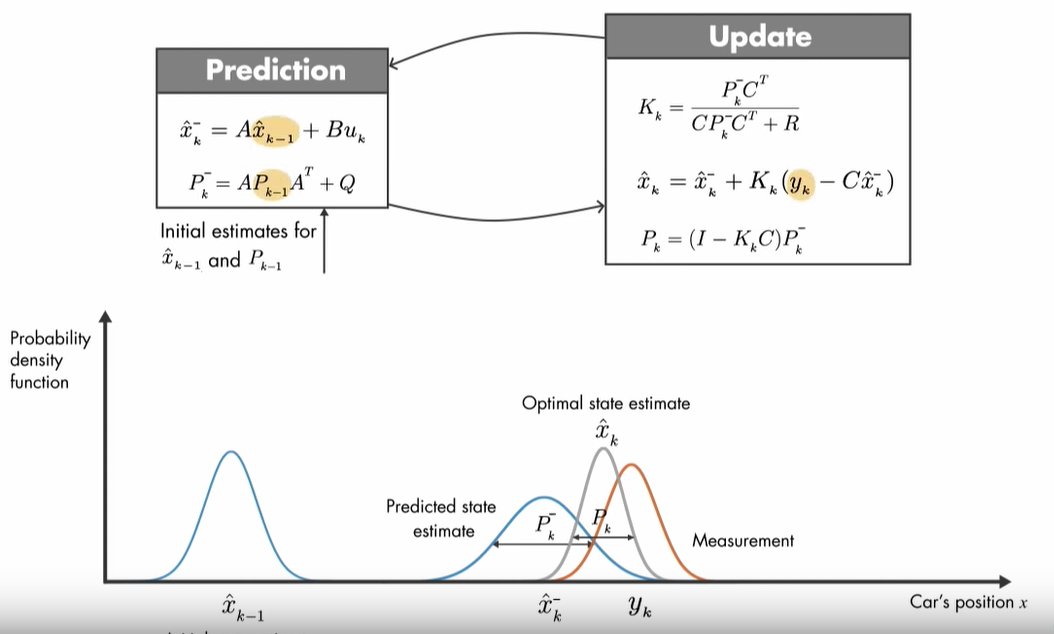


De exemplu daca Covarianta noastra tinde catre zero, atunci vom avea C^-1 ( care este = 1, in cazul acestui exemplu, adik 1 state).

Dar, In cazul in care our Prior error covariance, este aproape de zero:



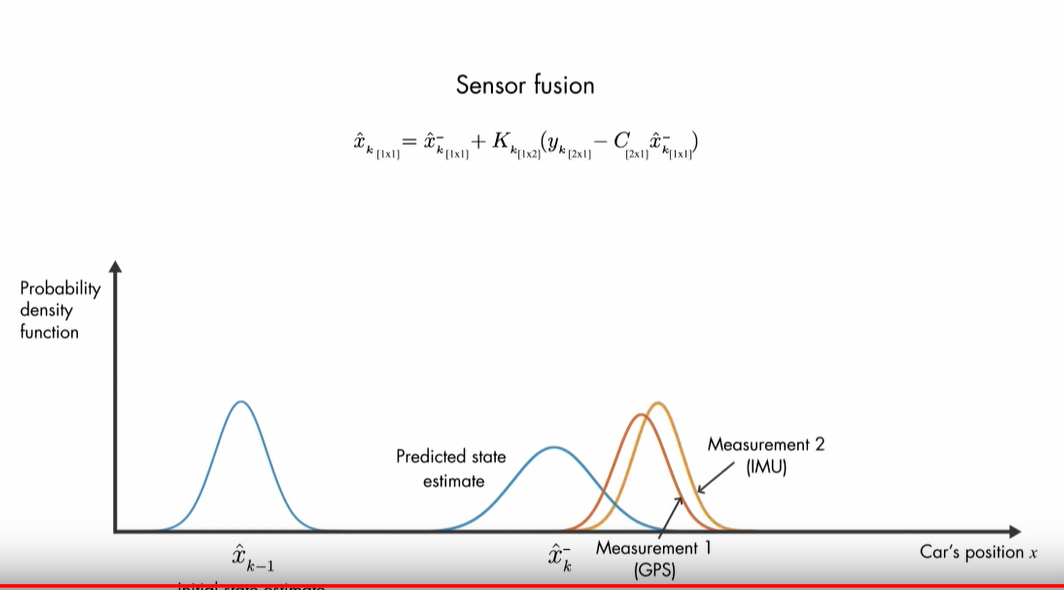
Once we’ve calculated the update equations, in the next time step, the posterior estimates are used to predict the new priori estimates. And the algoritm repreats itself. Notice that to estimate the current state, the algorithm doesn’t neet all the past information. It only needs the estimated state and error covariance matrix from the previous time step and the current measurement.



This is what makes the Kalman filter recursive. Now you know the set of equations needed to implement the Kalman filter algorithm.

Kalman filter = Sensor fusion algorithm

If you have two measurements, the dimensions y, C, and K, matrices would change as shown here



* This time we will multiply three probability density functions together to find the optimal estimate of the car’s position.