Introduction to Linear System Control

& Kalman Filter

arnaud.nauwynck@gmail

Outline

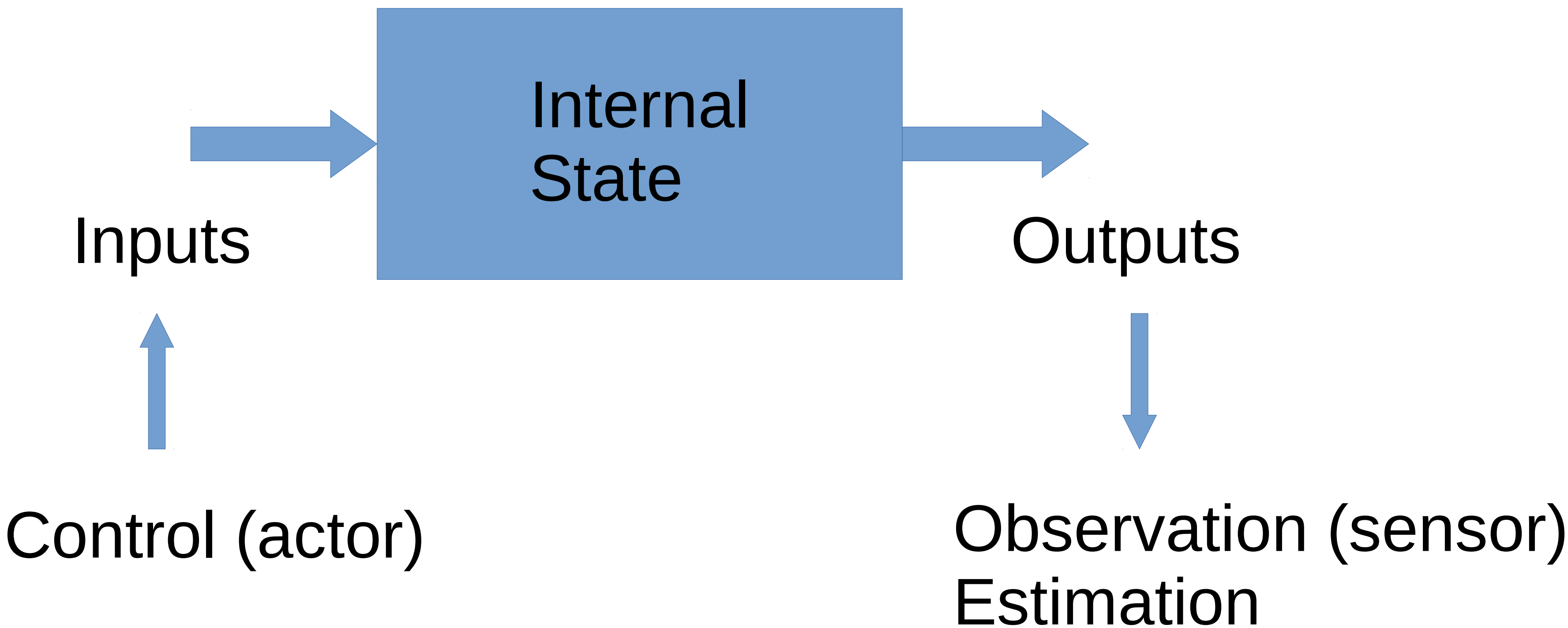
- System Model Equation

- Input = Theory of Control
  (Riccati Equation – Kalman Gain)

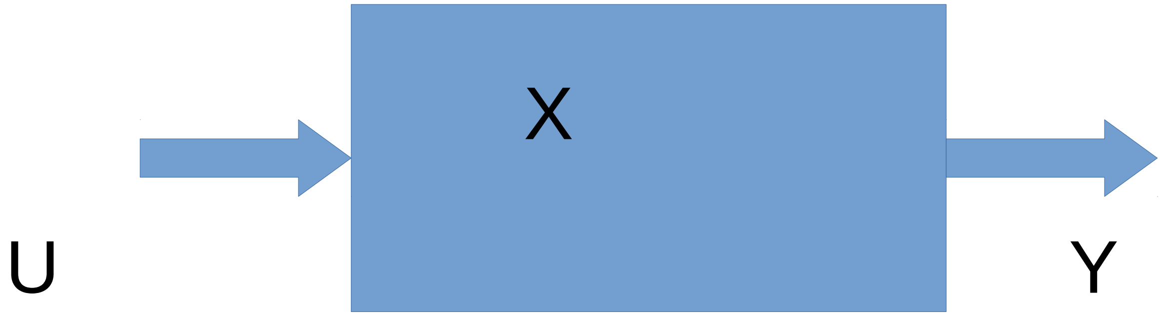
- Output = Theory of Estimation

  (Kalman Filter)

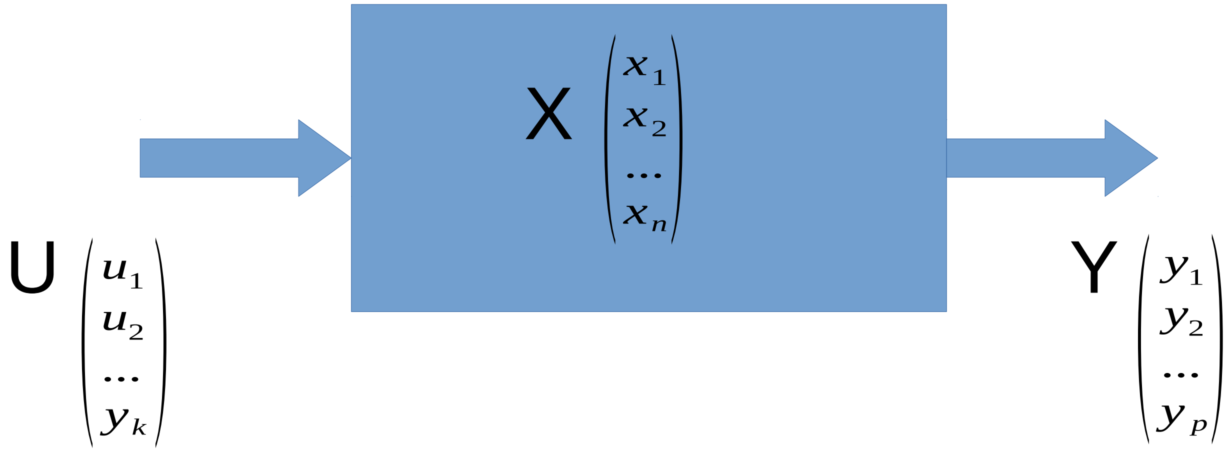
System … Black Box



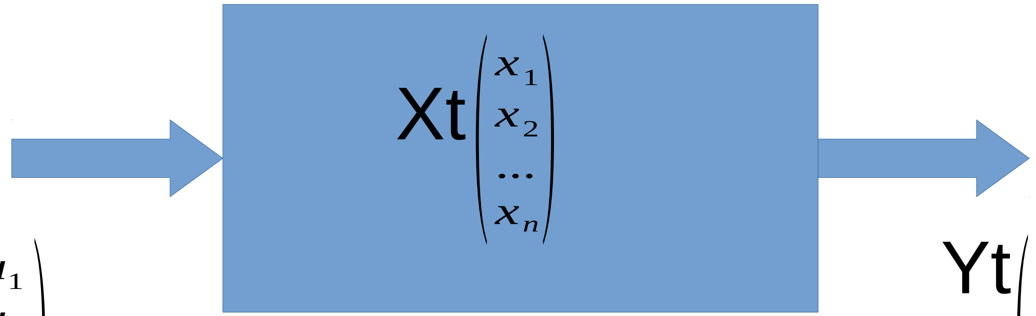
Notation



Dimension / Freedom Degrees



State Model Equation
Continuous case

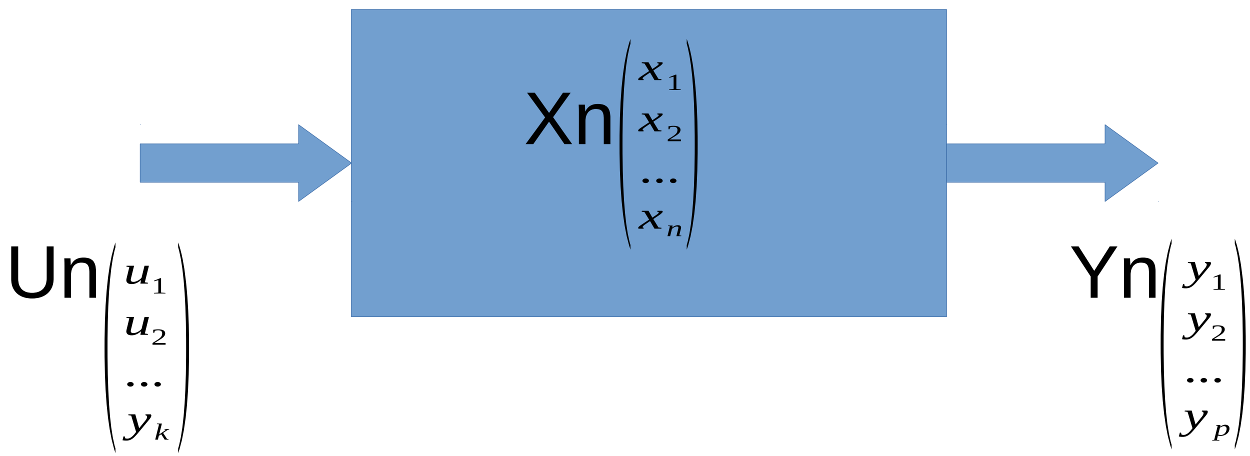


$$Xt\begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} \quad Ut\begin{pmatrix} u_1 \\ u_2 \\ \dots \\ y_k \end{pmatrix} \quad Yt\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_p \end{pmatrix}$$

$$\dot{X}_t = \frac{\partial X_t}{\partial t} = f\left(X_t, U_t, W_t\right)$$

with Wt $\begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{pmatrix}$: white noise (brownian)

State Model Equation
Discrete case



$$X_{n+1} = f\left(X_n, U_n, W_n\right)$$

with Wn $\begin{pmatrix} w_1 \\ w_2 \\ \cdots \\ w_n \end{pmatrix}$: white noise (normal / gaussian)

Time Discretisation

$$t_{n+1} = t_n + dt$$

$$\dots t_n = n \,.\, dt + t_0$$

$$X_n = X(t_n)$$

$$\dots \dot{X}_t \,.\, dx = X(t_{n+1}) - X(t_n) = X_{n+1} - X_n$$

Note on CPU Timers : for fixed delay => use Real Time linux or hardware …
Typical loop timer: 10ms

State Model Differential Equation Order N => Order1 - Dimension N

$$\frac{\partial^n X_t}{\partial^n t} = f\left(\frac{\partial^{n-1} X_t}{\partial^{n-1} t}, \frac{\partial^{n-2} X_t}{\partial^{n-2} t}, \ldots, X_t\right)$$

let $X_t = \begin{pmatrix} X_t \\ \frac{\partial X_t}{\partial t} \\ \frac{\partial^2 X_t}{\partial^2 t} \\ \vdots \\ \frac{\partial^{n-1} X_t}{\partial^{n-1} t} \end{pmatrix}$ then $\dot{X}_t = \begin{pmatrix} \frac{\partial X_t}{\partial t} \\ \frac{\partial^2 X_t}{\partial^2 t} \\ \vdots \\ \frac{\partial^n X_t}{\partial^n t} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & \cdots & \\ 0 & 0 & 1 & \cdots & \\ \vdots & & & \ddots & 0 \\ 0 & & & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X_t \\ \frac{\partial X_t}{\partial t} \\ \vdots \\ \frac{\partial^{n-1} X_t}{\partial^{n-1} t} \end{pmatrix} + \begin{pmatrix} 0 \\ \vdots \\ f(\ldots, X_t) \end{pmatrix}$

Which can be written as $X_t = A.X_t + B.U_t$

where A=matrix(n,k) B=matrix(1,k) Ut=vector(k)

Example Order 2
Newtown Mecanic

$$m.\vec{a}=\sum \vec{F}$$

Write with x,y coord:

$$\begin{pmatrix} m\ddot{x}=f_x \\ m\ddot{y}=f_y \end{pmatrix}$$

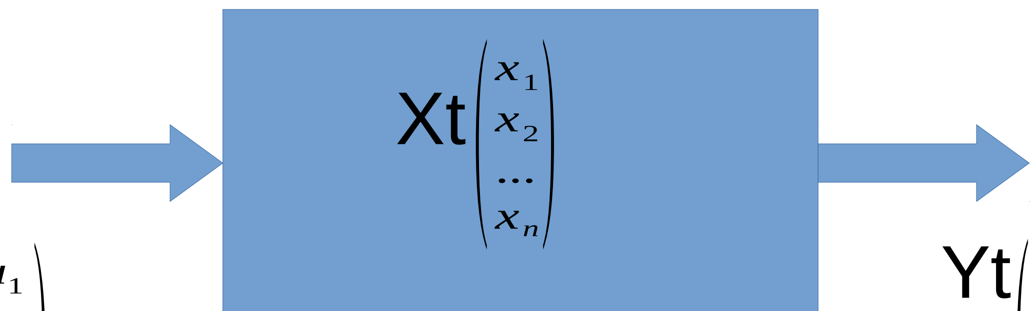
using:

$$X_t=\begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix} \quad A=\begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad B=\begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \quad U=\begin{pmatrix} f_x/m \\ f_y/m \end{pmatrix}$$

Finally:

$$\dot{X}_t=A.X_t+B.U_t$$

Linear State Model Equation
Continuous case



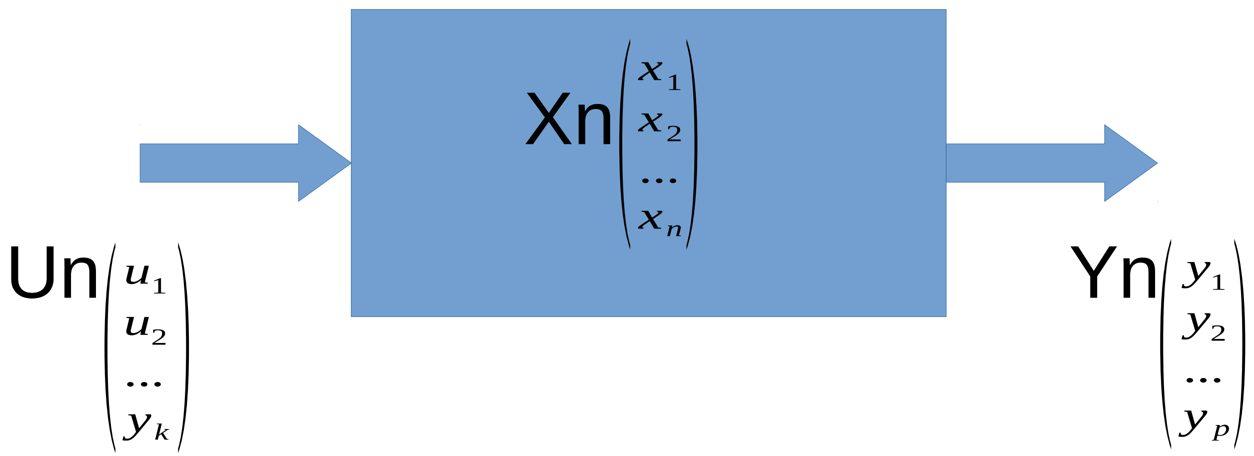
Ut $\begin{pmatrix} u_1 \\ u_2 \\ \dots \\ y_k \end{pmatrix}$

Yt $\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_p \end{pmatrix}$

$$\dot{X}_t = A.X_t + B.U_t + D W_t$$

with Wt $\begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{pmatrix}$: white noise (brownian)

Linear State Model Equation
Discrete case



$$X_{n+1} = A.X_n + B.U_n + D.W_n$$

With Wn $\begin{pmatrix} w_1 \\ w_2 \\ \dots \\ w_n \end{pmatrix}$: white noise (normal / gaussian)

Outline

- System Model Equation

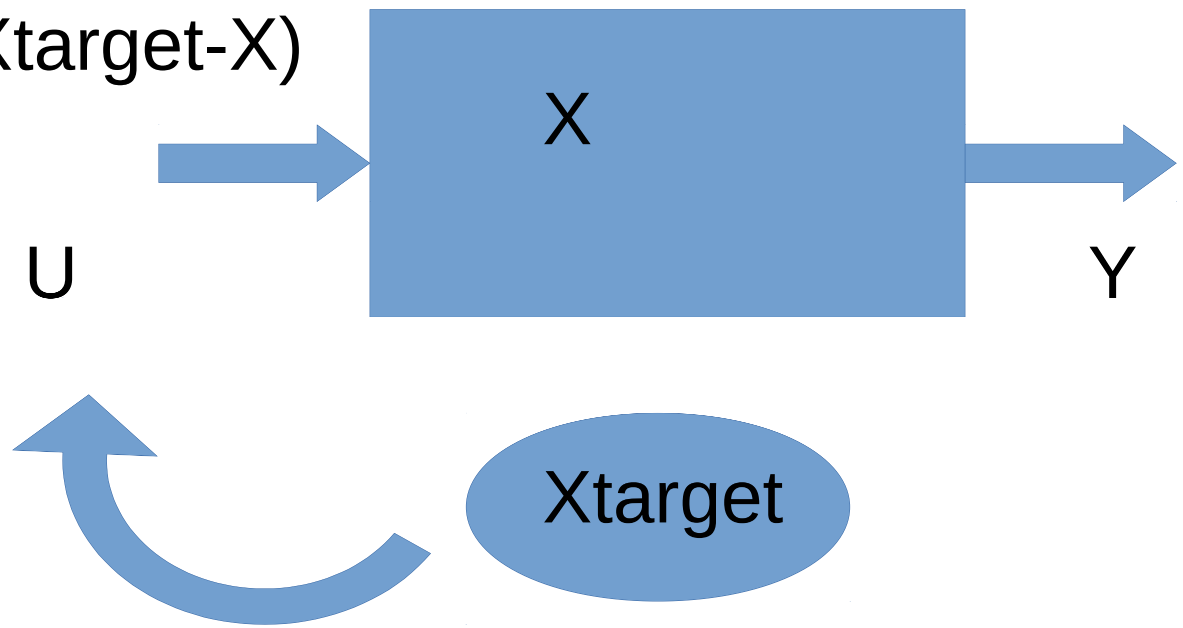
- Input = Theory of Control
  (Riccati Equation – Kalman Gain)

- Output = Theory of Estimation

  (Kalman Filter)

Feedback Loop : Follow Trajectory

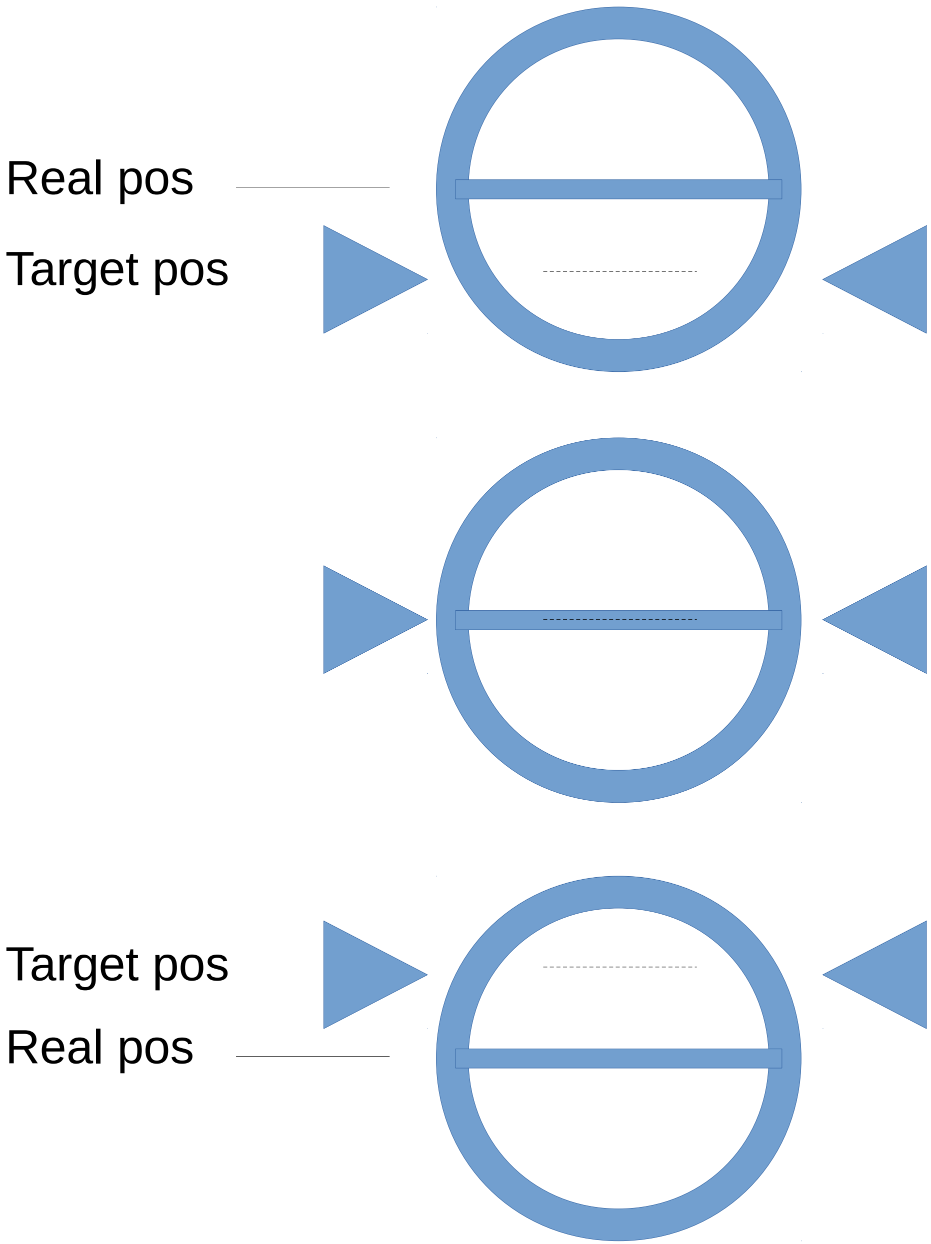
U = -K(Xtarget-X)



Xtarget-X = target error to correct

K = feedback gain (= kalman control gain)

Example



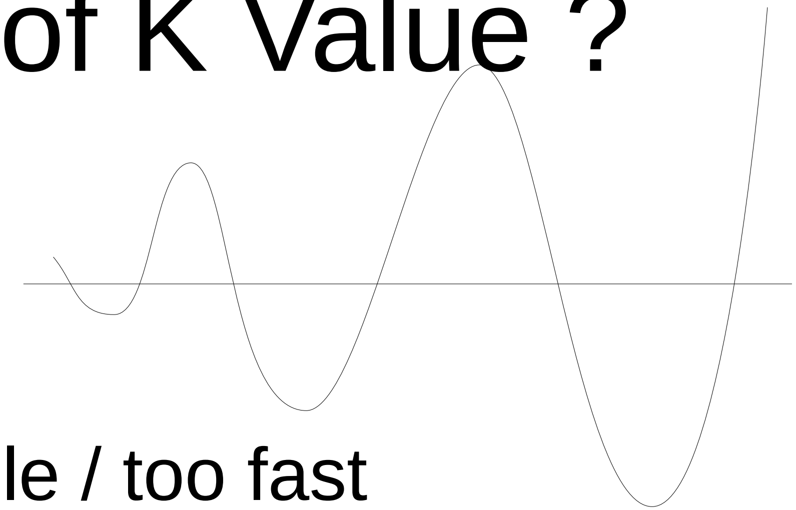
Too HIGH => GO Down

OK => don't move

Too LOW => GO Up

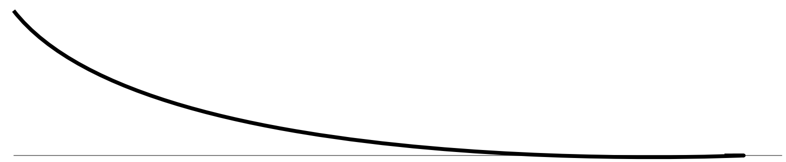
Choice of K Value ?

K Too BIG
(K > Kmax)

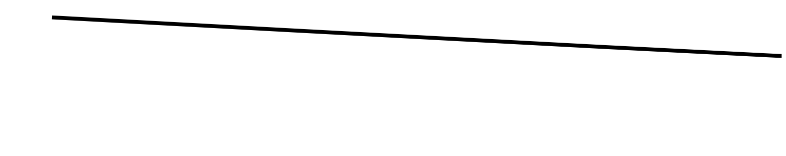


=> Divergent / unstable / too fast

K Optimal?
(Kmin < K < Kmax)



K Too SMALL
(0 < K < Kmin)



=> Not convergent / not reactive / too slow

Response Time Shift

Optimal in Theory :

Ut = -K (Xtarget - Xt)

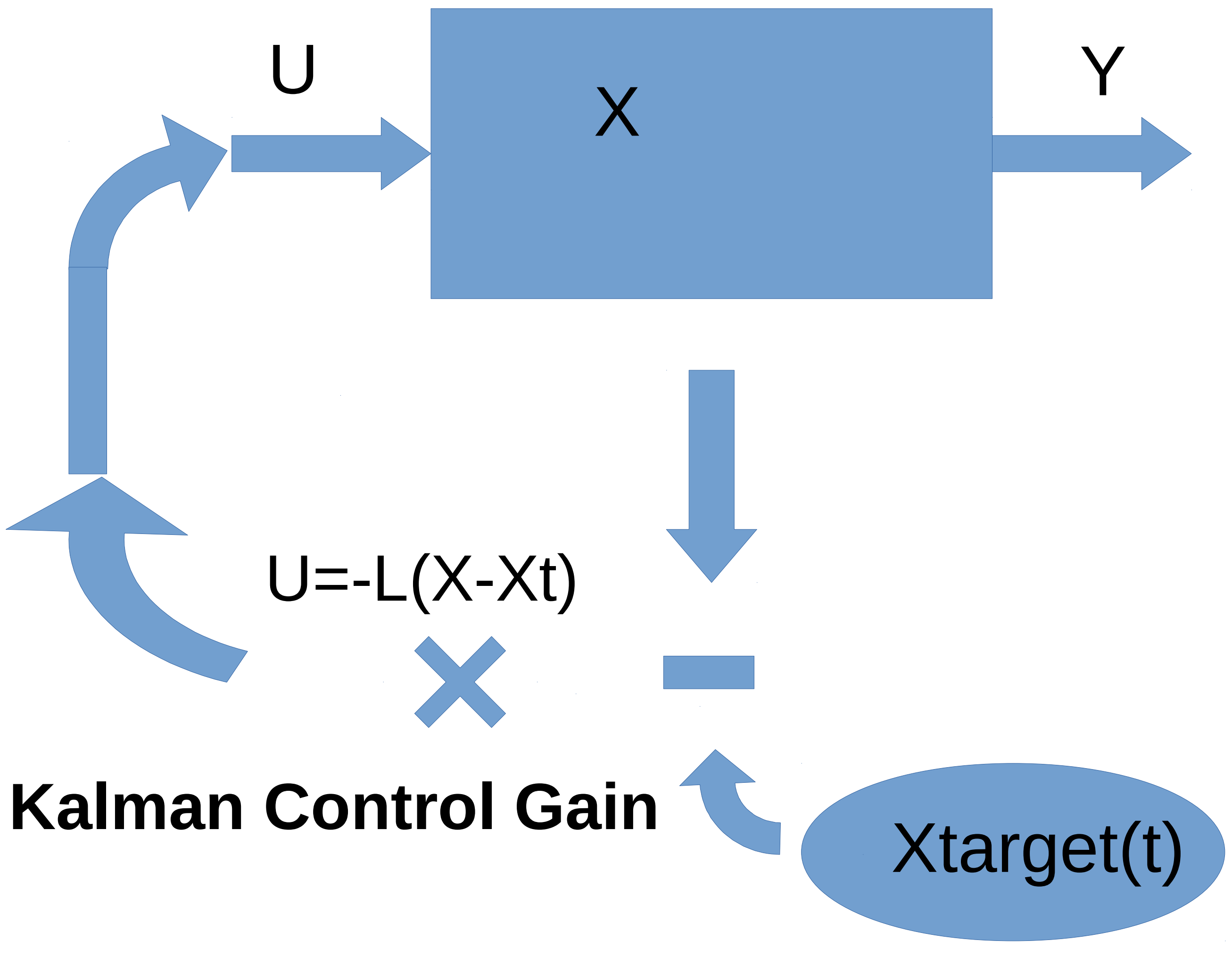
In practise ...

Ut+delayt = -K (Xtarget - Xt)

If delayt too big:
response time of computation > typical time of system
Then instability, vibration..
(example: you are drunk … reaction > 100ms .. don't drive)

Achieve smallest delayt
Maybe split K1, K2...Kn with
K1= in hardward=nanos / K2: 10ms / K3: 100ms ...

Control Gain
System Drawing



Computation for Optimal K ?

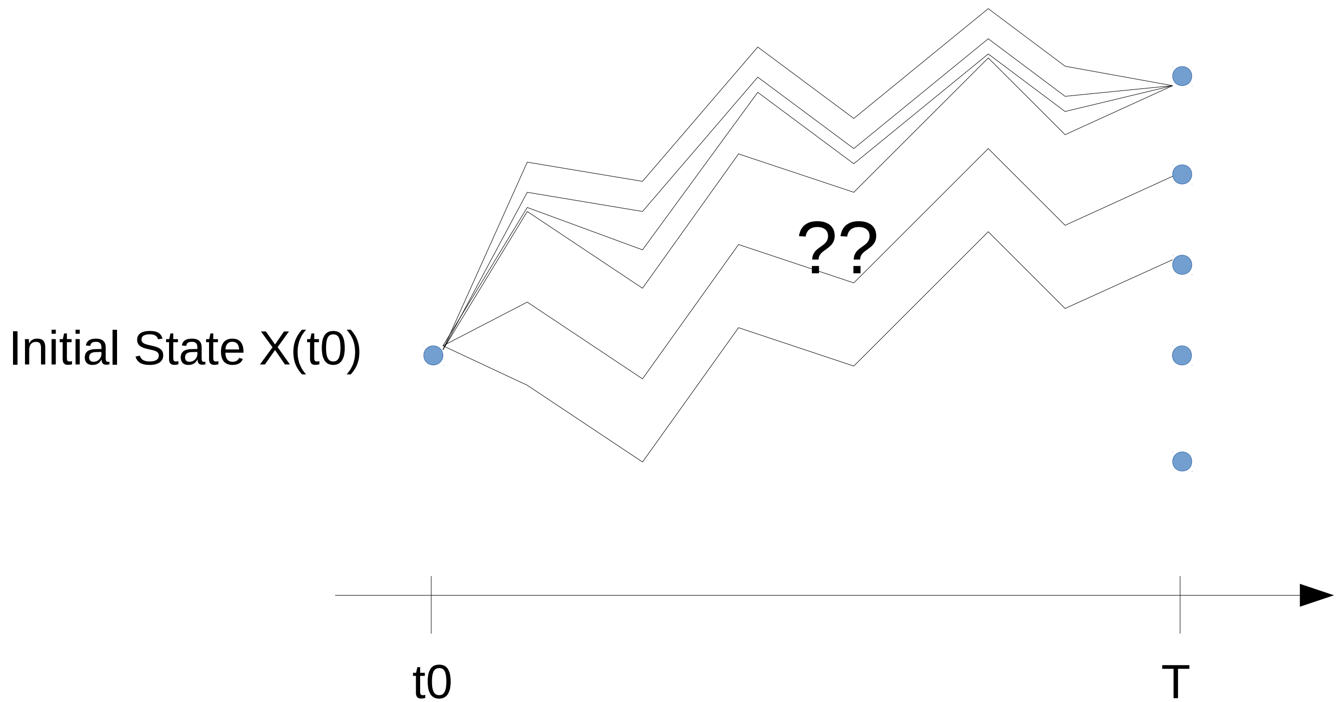
$$V(trajectory)=\int_{t_0}^{T} {}^t(X_t - Xtarget_t)\,Q\,(X_t - Xtarget_t) + {}^tU_t\,R\,U_t\,dt$$

Term for cumulated error of trajectory following

Term for cumulated energy of control

Choose 2 symmetric matrices Q=(n,n) & R=(k,k)

Compute for All Trajectories??

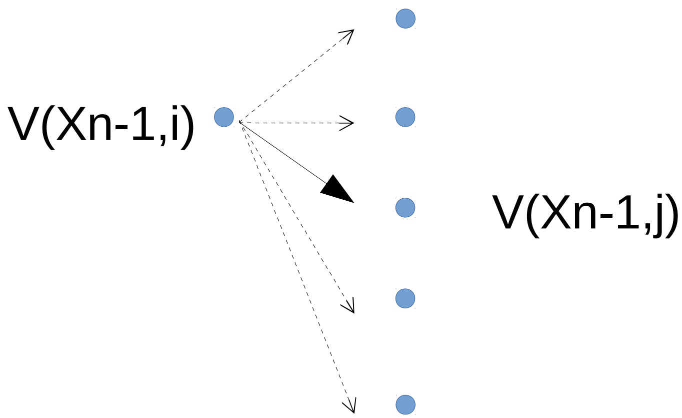


Dynamic Programming 1/3

Principle:
If X0 → X1 → X2 → …. Xn is the optimal trajectory from X0 to any Xn
Then Xi → Xi+1 → … Xn is the optimal (sub) trajectory from Xi to any Xn

Compute => Xn-1 → Xn the optimal last step for Xn-1 to any Xn

Computation for last Step N:
Foreach K, compute
V(Xn-1,i) = arg min j cost(Xn-1,i → Xn,j) + V(Xn,j)

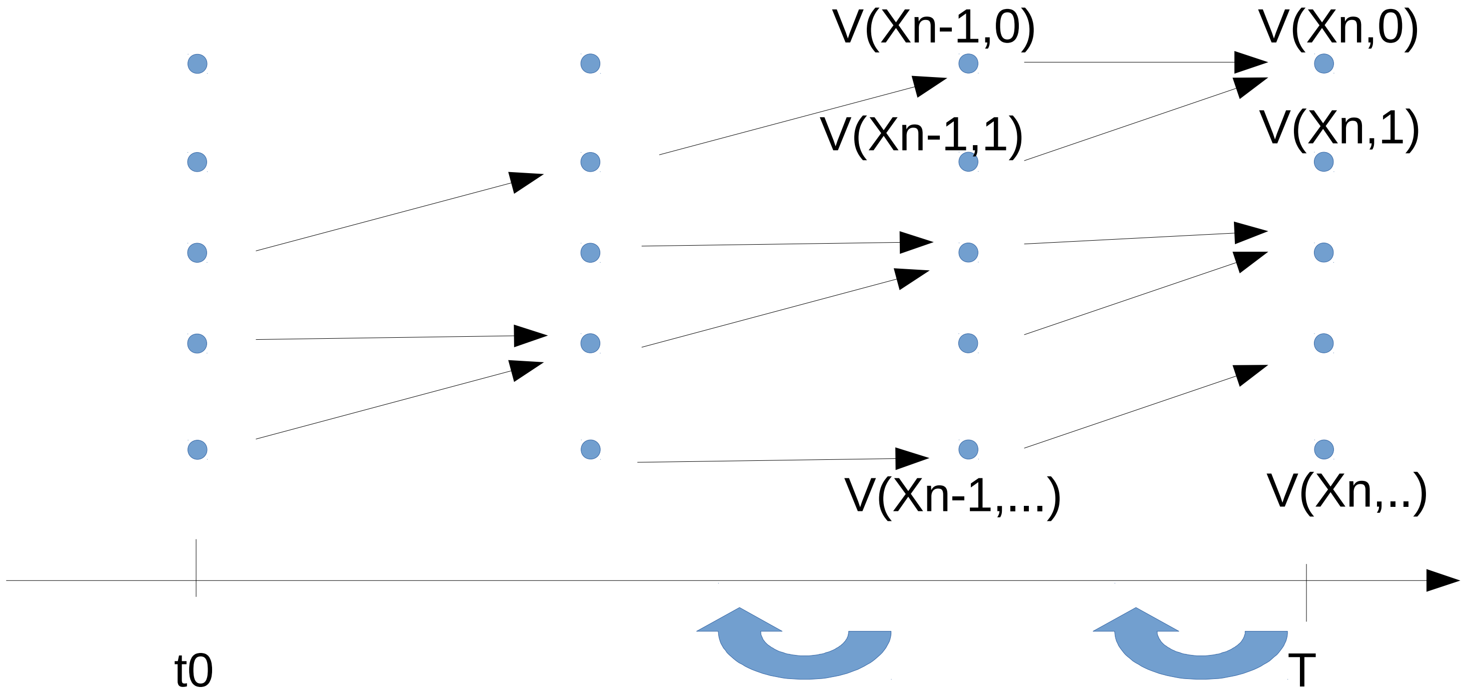


Algorithm similar to
Bellman-Kalaba “shortest path”
To any destinations

Dynamic Programming 2/3

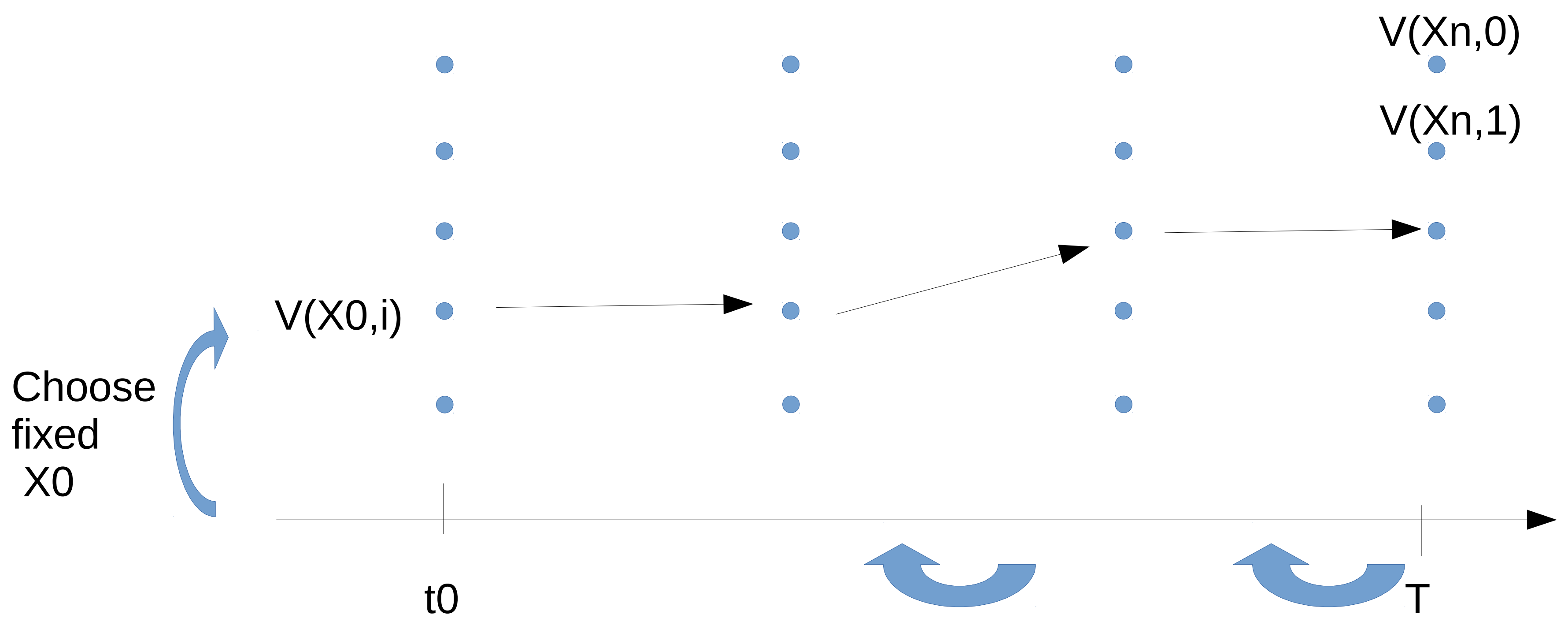
Recurse … to get optimal Trajectories from any points:

- initialize V(Xn,i)
- recurse n → n-1 :   compute V(Xn-1,i) … remember direction from Xn-1,i → Xn



Dynamic Programming 3/3

Pick up optimal trajectory from t0
(remembering each step directions Xn-1,i → Xn)



Application Computation of Optimal Kalman Gain

WIKIPEDIA
The Free Encyclopedia

Article Talk Read Edit View history Search Wikipedia

Algebraic Riccati equation

From Wikipedia, the free encyclopedia

An **algebraic Riccati equation** is a type of nonlinear equation that arises in the context of infinite-horizon optimal control problems in continuous time or discrete time.

A typical algebraic Riccati equation is similar to one of the following:

the continuous time algebraic Riccati equation (CARE):

$$A^T X + XA - XBR^{-1}B^T X + Q = 0$$

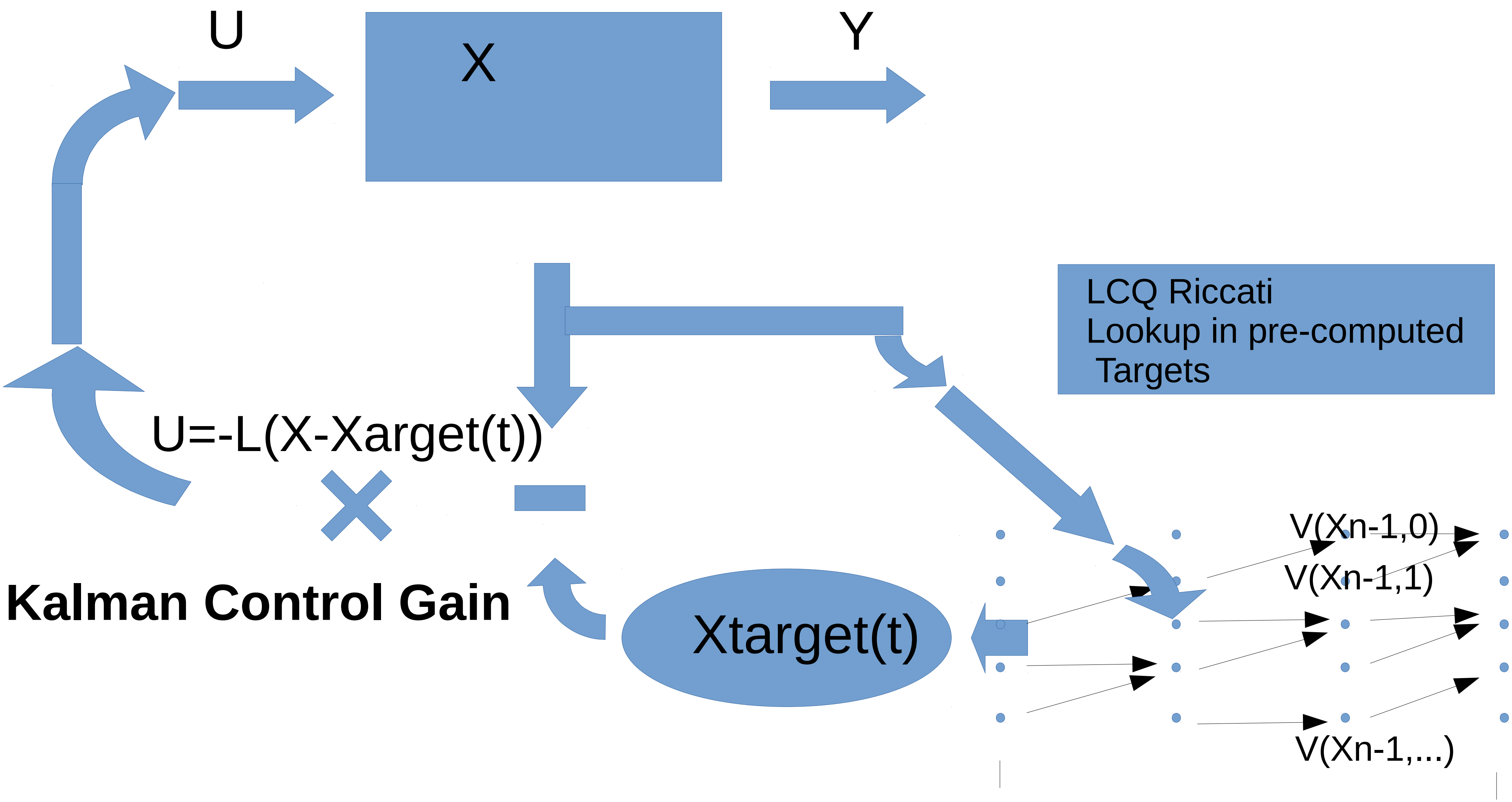
or the discrete time algebraic Riccati equation (DARE):

$$X = A^T XA - (A^T XB)(R + B^T XB)^{-1}(B^T XA) + Q.$$

X is the unknown n by n symmetric matrix and A, B, Q, R are known real coefficient matrices.

It is called Riccati Equation (but nothing to do with Italian Mathematician)
This is from Kalman !

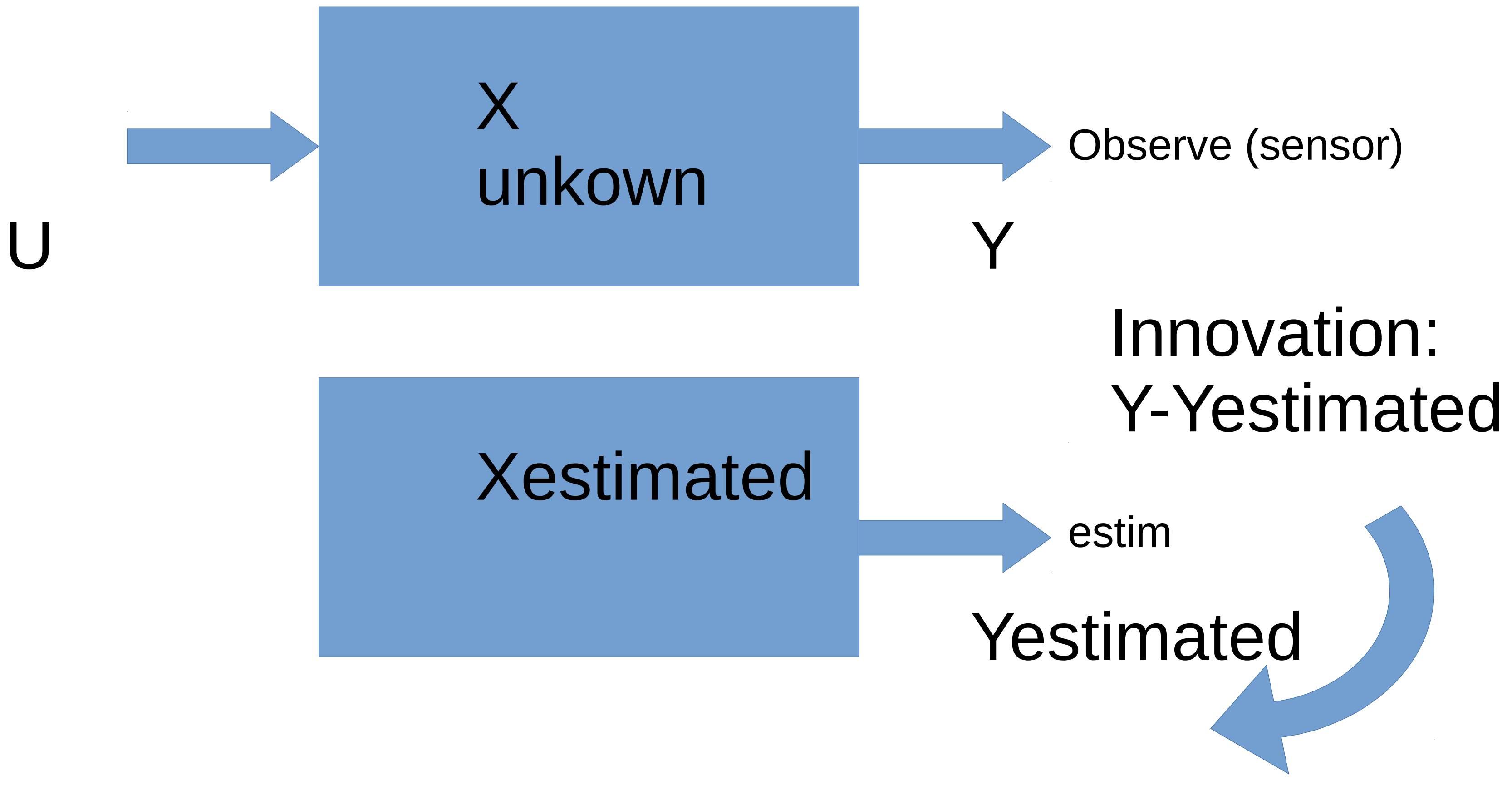
Control Gain + LCQ Riccati System Drawing



Outline

- System Model Equation
- Input = Theory of Control
  (Riccati Equation – Kalman Gain)
- Output = Theory of Estimation
  (Kalman Filter)

Estimation Feedback Loop from Observations



Dimensions / Observation vs Degrees of Freedom

$$Xn \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} \qquad Yn \begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_p \end{pmatrix}$$

In general n > p : not every variables are observables

You mesure only a projection

Example 1: $Xn \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ $Yn \begin{pmatrix} x \\ y \end{pmatrix}$ In 3D word ...only seing 2D images

Example 2: $Xn \begin{pmatrix} x \\ \dot{x} \\ \ddot{x} \end{pmatrix}$ $Yn \begin{pmatrix} \ddot{x} \end{pmatrix}$ On your cell-phone, you only have an accelerometer

Redundant Sensors Measures => merge for Accuracy

Same “measure” several times by different sensors
=> get different conflicting values for same variable !

=> use ponderations between sensors accuracy / speeds

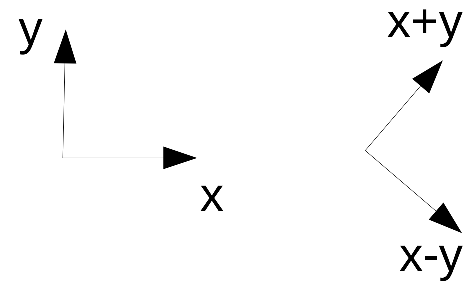
Example 1: $Xn \begin{pmatrix} x \\ y \end{pmatrix}$ $Yn \begin{pmatrix} x \\ x+1 \\ 2x \\ 3x-1 \end{pmatrix}$

Example: Take several Pictures
(with different expositions)
=> merge pictures to remove noise

Deduce State from Linear Combination of Sensors ?

Example 1: Xn $\begin{pmatrix} x \\ y \end{pmatrix}$ Yn $\begin{pmatrix} x+y \\ x-y \end{pmatrix}$

Measure in different basis
=> change basis (inverse basis matrix)



Example 2: Xn $\begin{pmatrix} x \\ y \end{pmatrix}$ Yn $\begin{pmatrix} dist_{satellite\,1} \\ dist_{satellite\,2} \\ dist_{satellite\,3} \\ dist_{satellite\,4} \end{pmatrix}$

GPS :
Compute latitude/longitude position
from distances to Satellites...
in your cell-phone also

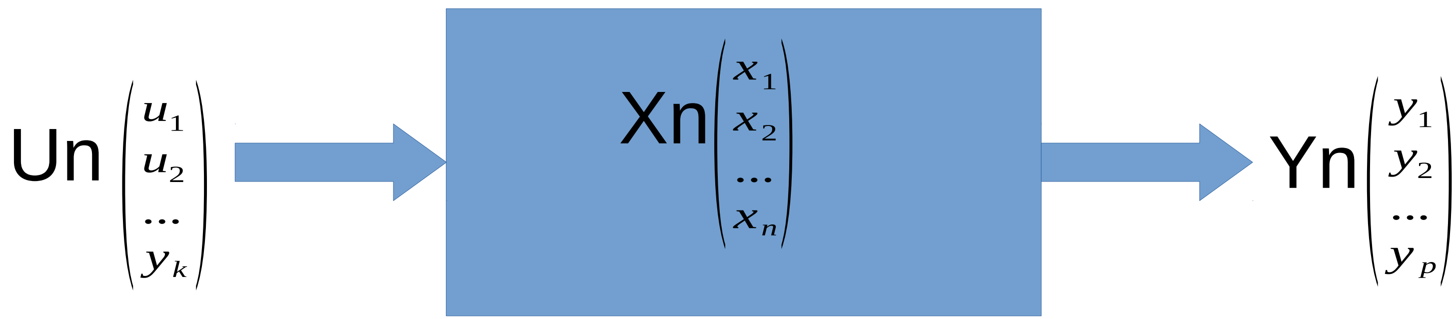
Deduce State From Current Sensor + N Past Sensor Measures

Mathematical Theorem on Linear Observable System …

if ($AC, A^2C, A^3C, ..A^nC$) has dimension N
=> then system is observable

Notice Transposition:
if ($AB, A^2B, A^3B, ..A^nB$) has dimension N
=> then system is controlable

Observation Model Equation
Continuous / Discrete case



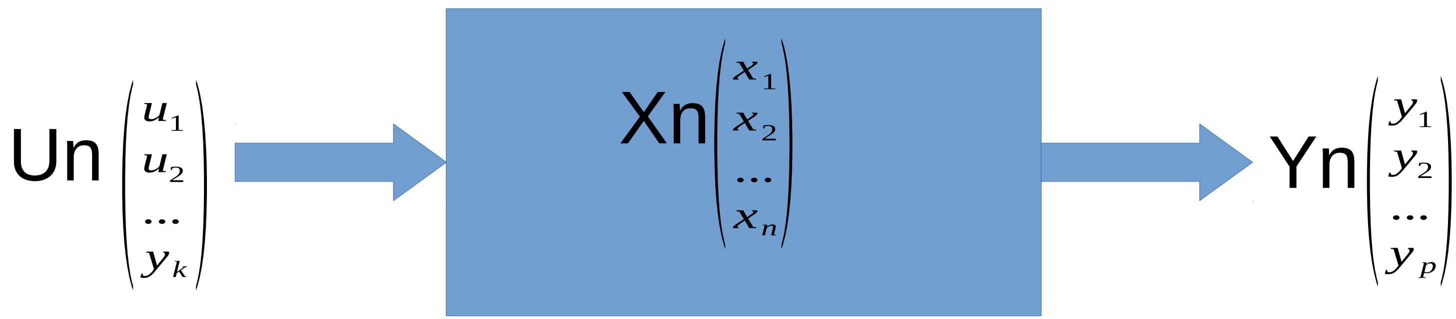
Discrete case : $$Y_n = f(X_n, U_n, W_n)$$

with Wn $\begin{pmatrix} w_1 \\ w_2 \\ \ldots \\ w_n \end{pmatrix}$: white noise (normal / gaussian)

Continuous case : $$Y_t = f(X_t, U_t, W_t)$$

with Wt $\begin{pmatrix} w_1 \\ w_2 \\ \ldots \\ w_n \end{pmatrix}$: white noise (brownian)

Linear Observation Model Equation
Continuous / Discrete case



C=matrix(p,n), F=matrix(p,k) in general=0, E=matrix(p,p)

Discrete case :

$$Y_n = C\,X_n \left(+F\,U_n\right) + E\,W_n$$

with Wn $\begin{pmatrix} w_1 \\ w_2 \\ \cdots \\ w_k \end{pmatrix}$: white noise (normal / gaussian)

Continuous case :

$$Y_t = C\,X_t \left(+F\,U_t\right) + E\,W_t$$

with Wt $\begin{pmatrix} w_1 \\ w_2 \\ \cdots \\ w_k \end{pmatrix}$: white noise (brownian)

Sensor (so Measures) have many Weaknesses !

White Noise : random additive noise each time you measure

Other weaknesses :

Model Error … what you measure is not really what you think, your equation are wrong

Biased Noise : the average measure is shifted compared to real
you need to re-calibrate your sensor

Discrete Sampling Time : your eletronic sensor is not very fast ...example every 10ms

Sampling Not Strictly Regular : every 10ms? but sometime 9ms, sometime 11ms

Integer Rounding (=Quantification) : your Analog To Digital converter is only precise
to 10 bits for instance (= 1024 values)

Time Delay : you get results 5ms late after they are really measured

Missing/Irregular Measures Arrival : sometime you don't have measure,
or measures are not regular at all (example: satellites)

Handling Irregular Measures

Example of handling “Missing/Irregular Measures Arrival”

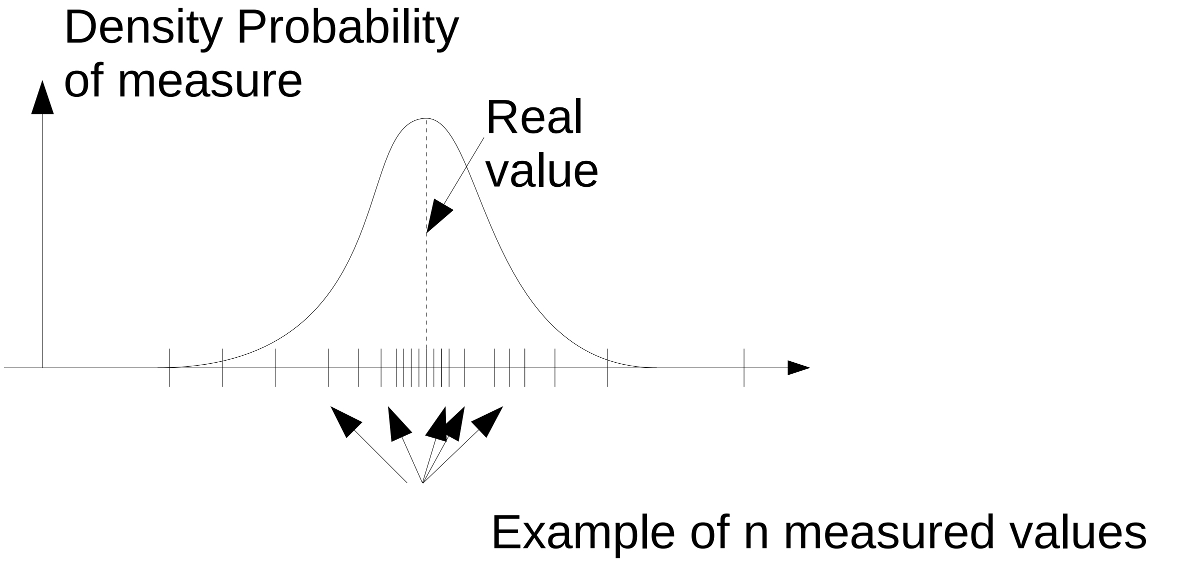
When receiving measures :

measure set1 => then use equation1 $$Y_n = C_1 X_n + E_1 W_n$$

measure set2 => then use equation2 $$Y_n = C_2 X_n + E_2 W_n$$

measure set3 => then use equation3 ...

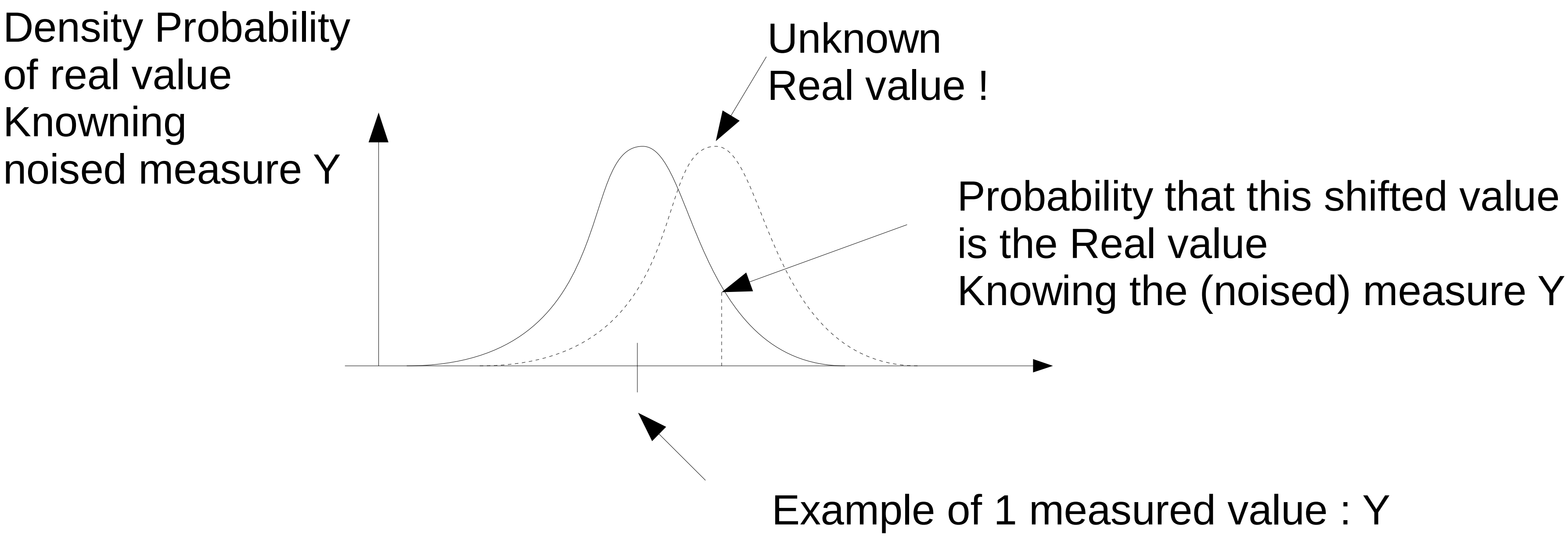
Measures Noise term: E Wn



(White) Noise =
repeating N measures give N different values following a Gaussian distribution

Biased = when the average is shifted

Noise Measure => Real Value?



Merge N noised measures : Average

$$Average(Y_{1..n}) = \frac{1}{N}\left(Y_1 + Y_2 + \ldots Y_n\right)$$

If all measures Yi have precisions $\pm\sigma$

Then Average(Y) has precision $\pm\frac{\sigma}{\sqrt{N}}$

(Theorem called “Law of Big Numbers”, for gaussians distributions)

Probability Espectation (synonym: Average, Mean)

E(Y) = Expected of Y = average value of Y
… for all random events
weigthed by their probabilities

Discrete case: given (Y1,P1), (Y2,P2)...(Yn,Pn)
where Yi : value of event i with probability Pi,

$$E(Y) = \sum_i P_i Y_i$$

Continuous case: given density probability (Y(x),P(x)dx)
where Yx value for x<.<x+dx with probability P(x)

$$E(Y) = \int_{x=-\infty}^{+\infty} Y(x)\, P(x)\, dx$$

Probability Variance
(=centered moment of order 2)

V(Y) = Variance of Y = weighted average value of
square of centered (Y-E(Y))
… for all random events

$$Var(Y) = \sum_i P_i \bar{Y}_i^{\,2} \ldots where\ \bar{Y}_i = Y_i - E(Y)$$

$$Var(Y) = \int_{x=-\infty}^{+\infty} Y\overline{(}x)^2 P(x)\, dx$$

Var is moment of order 2
E(Y) was moment of order 1
… we can also define moment 3, 4 ...

Standard Deviation
(= square root of variance)

By definition:

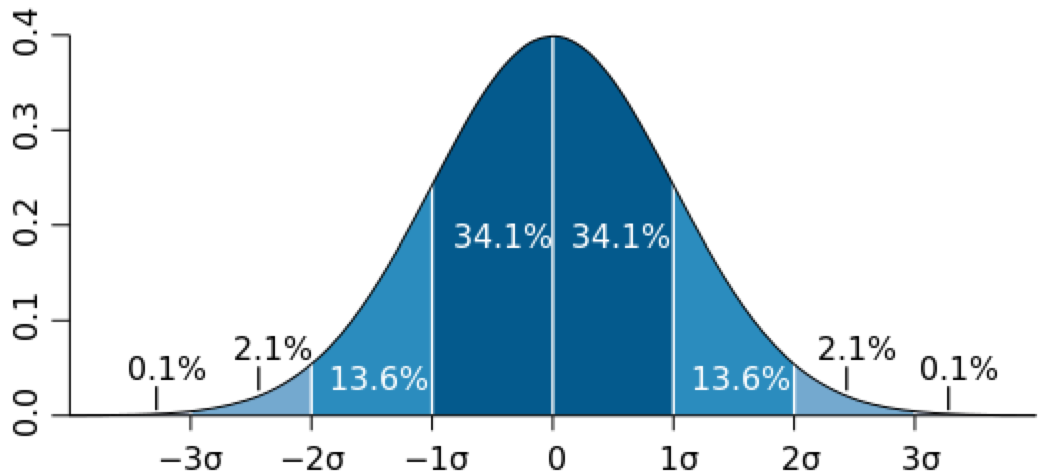
$$Var = \sigma^2$$

Equivalent:

$$\sigma = \sqrt{Var}$$

Standard Deviation of X has the same unit as X, E(X)
(if X is a distance, stddev is also distance)

Standard Deviation for Normal Distribution (Gaussian)



Probability ~68% that value in $[y-\sigma, y+\sigma]$

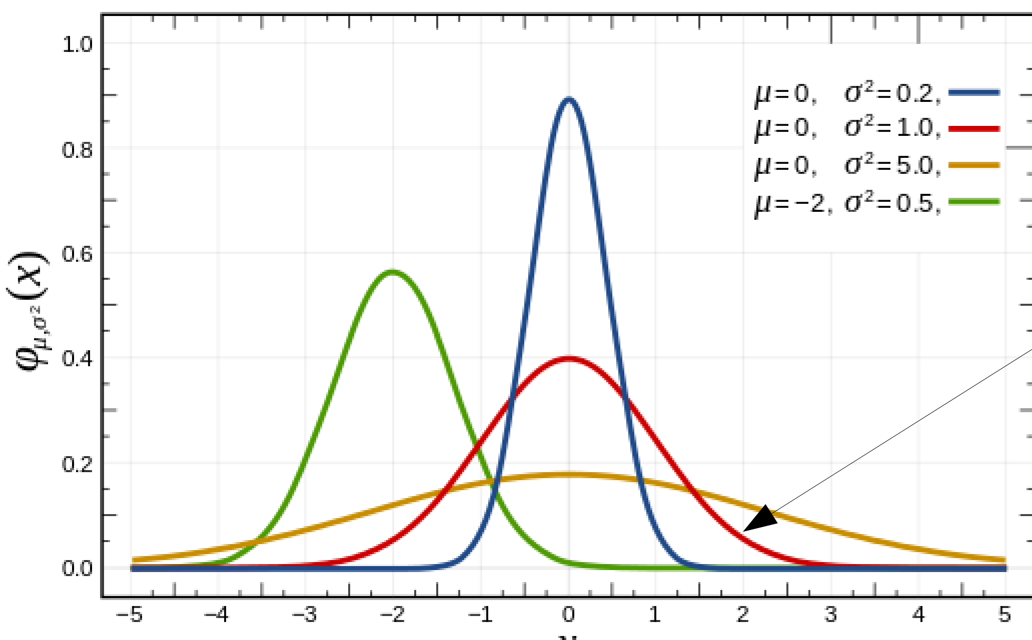
Probability ~95% that value in $[y-2\sigma, y+2\sigma]$

Probability ~99.7% that value in $[y-3\sigma, y+3\sigma]$

Gaussian Distribution
… defined by (Expect,StdDev)
Normale when StdDev=1

WIKIPEDIA
The Free Encyclopedia

$$f(x;\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



Normale when
$\sigma = 1$

Central Limit Theorem (~ Law of Big Numbers …)

Theorem:
Given N random independent measures Y1,Y2...Yn

Then centered average multiplied by $\sqrt{N}$
Converges to gaussian distribution $N\left(0,\sigma^2\right)$

$$\sqrt{n}\,\overline{Y}_{1..n} \rightarrow N(0,\sigma^2)$$

... For large enough N,
the N-average follow a gaussian $N(\mu,\frac{\sigma}{\sqrt{N}})$

Back To Measures Merge...

Sensor Merge point-of-view:

Suppose you have 2 measures Y1, Y2
with different stddeviations

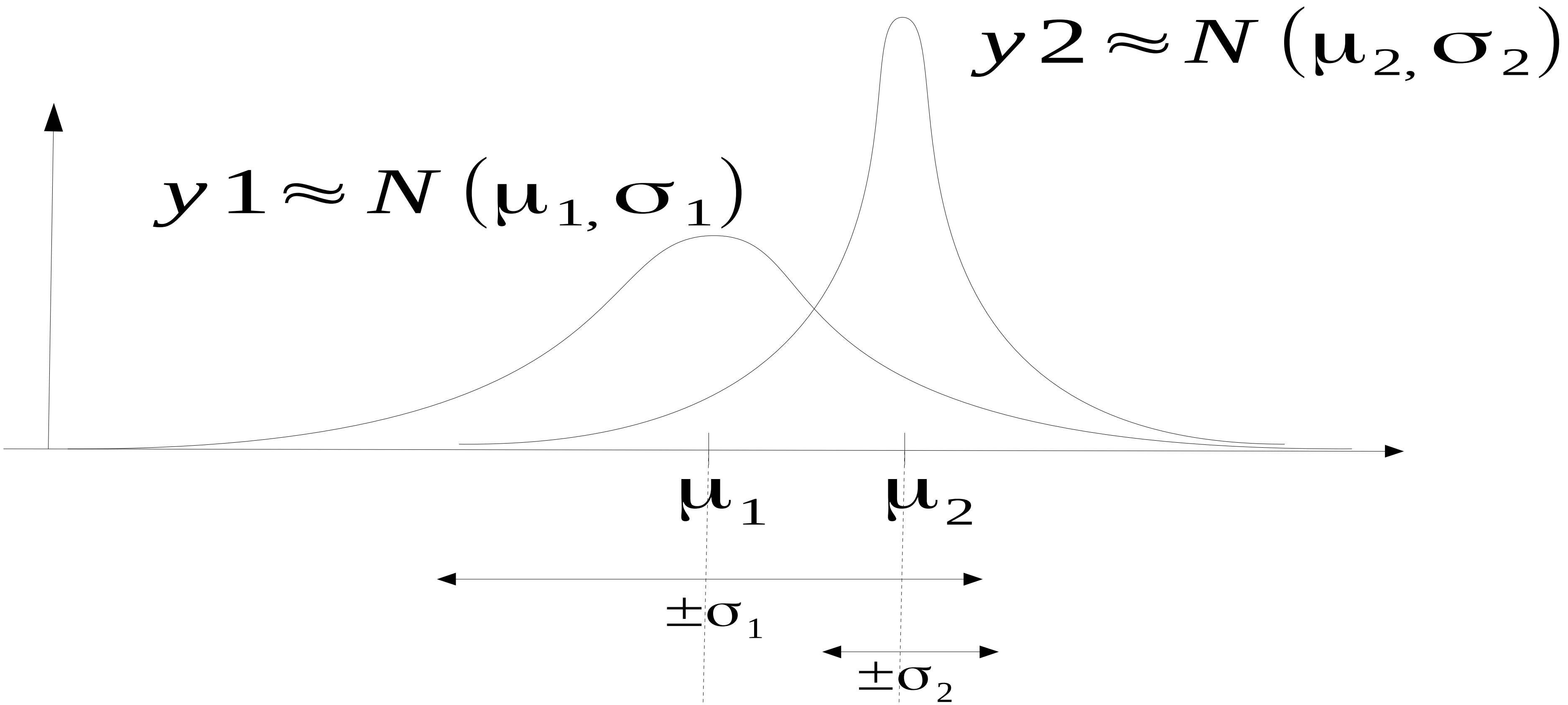
=> how to “weight-average” the 2 values ?

Kalman Filter point-of-view:

Suppose you have a-priori estimated prediction Y1
and a measure Y2

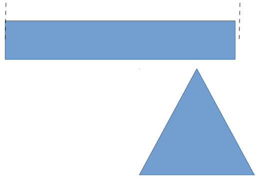
=> how to “weight-average” the 2 values ?

Merge 2 Gaussians ...



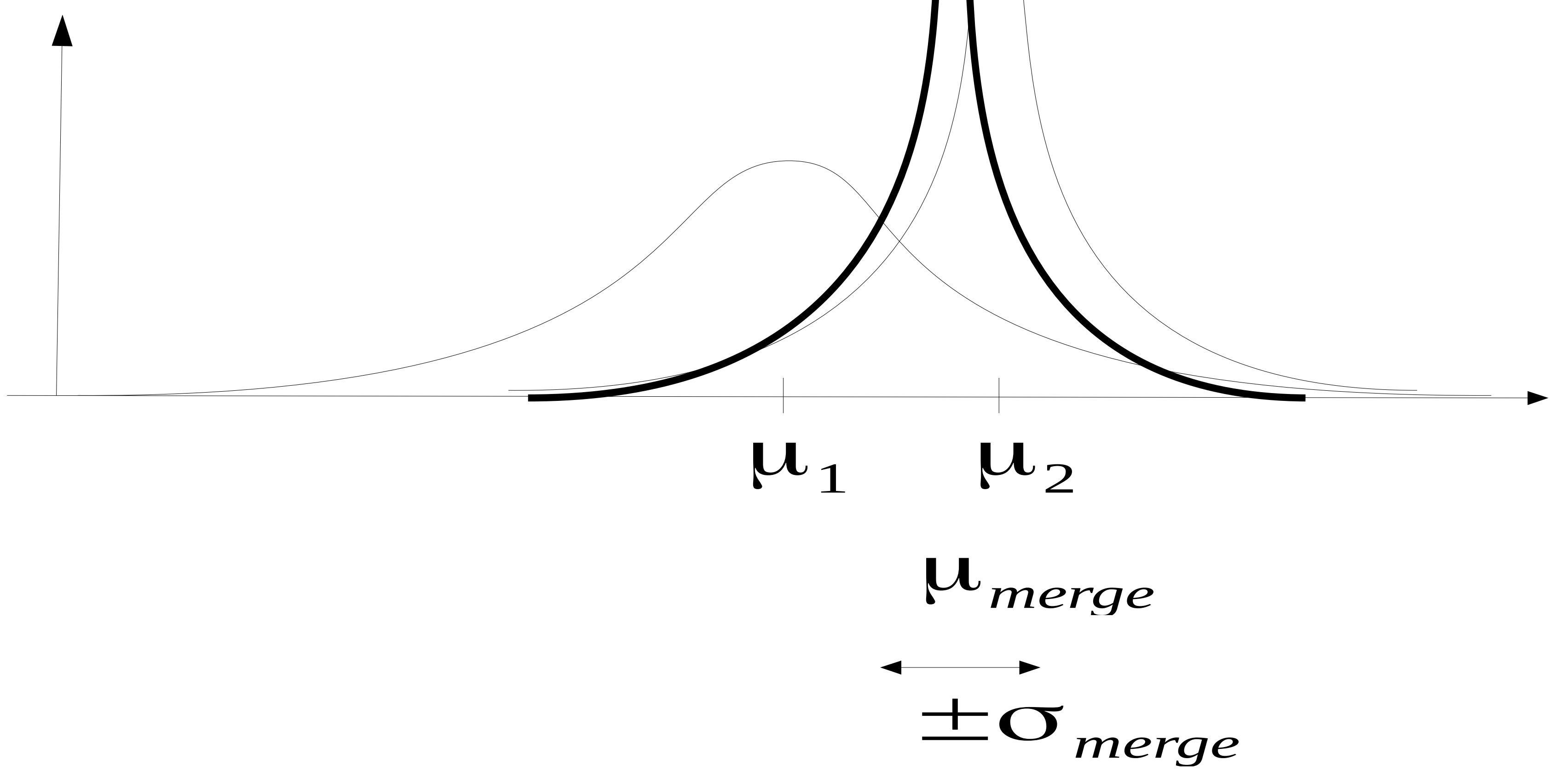
Merge Value MUST be in range $[\mu_{1,} \mu_2]$

Closest to the more accurate (smallest var)



Merge (multiply) 2 Gaussians => proportial to a Gaussian

$$y1 \approx N(\mu_{1,} \sigma_1)$$
$$y2 \approx N(\mu_{2,} \sigma_2)$$



Probability of 2 independent Events => Product of Probability

If A and B are two INDEPENDENT random events,
Then the probability of simultaneous event “A and B” is

$$P(A \wedge B) = P(A) . P(B)$$

Conditional probability (not independent): $P(A|B) = \frac{P(A \wedge B)}{P(B)}$

Y1 and Y2 are both dependent of Yreal

$$P(Y_1 \wedge Y_2 | Y_{real}) = P(Y_1 | Y_{real}) . P(Y_2 | Y_{real})$$

Computing Weigth for product of 2 Gaussians

Remember...

$$e^n = \underbrace{e.e...e}_{n} \qquad e^a.e^b = e^{a+b}$$

So

$$Y_1(x).Y2(x) = \left(\frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}}\right)$$

$$Y = k\,e^{-\frac{1}{2\sigma_1^2\sigma_2^2}\left(\sigma_2^2(x-\mu_1)^2+\sigma_1^2(x-\mu_2)^2\right)}$$

Quadratic form ax^2+bx+c

$$= (\sigma_1^2+\sigma_2^2)x^2 - 2(\sigma_2^2\mu_1+\sigma_1^2\mu_2)x + ...$$

Rewrite a(x+b/2a)² + ...

$$= (\sigma_1^2+\sigma_2^2)\left(x - \frac{\sigma_2^2\mu_1+\sigma_1^2\mu_2}{\sigma_1^2+\sigma_2^2}\right)^2 + ...$$

Result Gaussian Product

$$Y_1(\mu_{1,}\sigma_1).Y_2(\mu_{2,}\sigma_2)=cst.e^{-\frac{1}{2}\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}\left(x-\frac{\sigma_2^2\mu_1+\sigma_1^2\mu_2}{\sigma_1^2+\sigma_2^2}\right)^2}$$

$$=cst.Y(\mu,\sigma)$$

With

$$\mu=\frac{\sigma_2^2\mu_1+\sigma_1^2\mu_2}{\sigma_1^2+\sigma_2^2}$$

$$\frac{1}{\sigma^2}=\frac{\sigma_1^2+\sigma_2^2}{\sigma_1^2\sigma_2^2}$$

Re-interpret Expectation Weighted Sum

$$\mu = \frac{\sigma_2^2 \mu_1 + \sigma_1^2 \mu_2}{\sigma_1^2 + \sigma_2^2} \quad => \quad \mu = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \mu_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \mu_2$$

weight for u1 — Complementary weight for u2

Weigths are in [0, 1]
(0% to 100%)

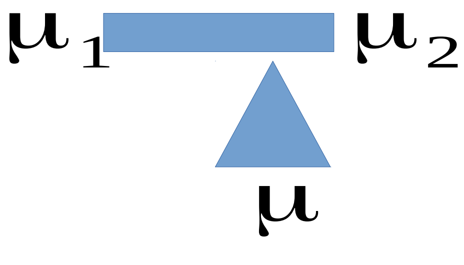
$$0 \le \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \le 1 \qquad 0 \le \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \le 1$$

Sum of both Weigths = 1
(sum is 100%)

$$\frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} = 1$$

Result value is in range $[\mu_1, \mu_2]$

Closest to the more accurate (smallest var)

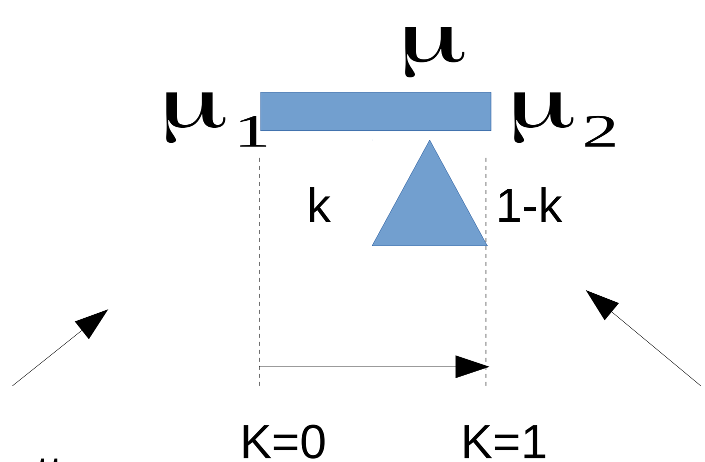


Re-interpret weight as Kalman Gain

posing $$K = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$

$$\mu = K\mu_1 + (1 - K)\mu_2$$

Choose Letter “K” in eternal memory of Rudolph E Kalman



When $\sigma_1 \ll \sigma_2$

Y2 not Accurate... Y1 better

=> more weight on Y1
K ~ 1   1-K ~ 0
=> closest to u1

When $\sigma_1 \gg \sigma_2$

Y1 not Accurate... Y2 better

=> more weight on Y2
K ~ 0   1-K ~1
=> closest to u2

Re-interpret StandardDeviation (variance, inverse of precision)

$$\frac{1}{\sigma^2} = \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 \sigma_2^2}$$

=>

$$\sigma^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$

Also see as Sum of inverse of square

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}$$

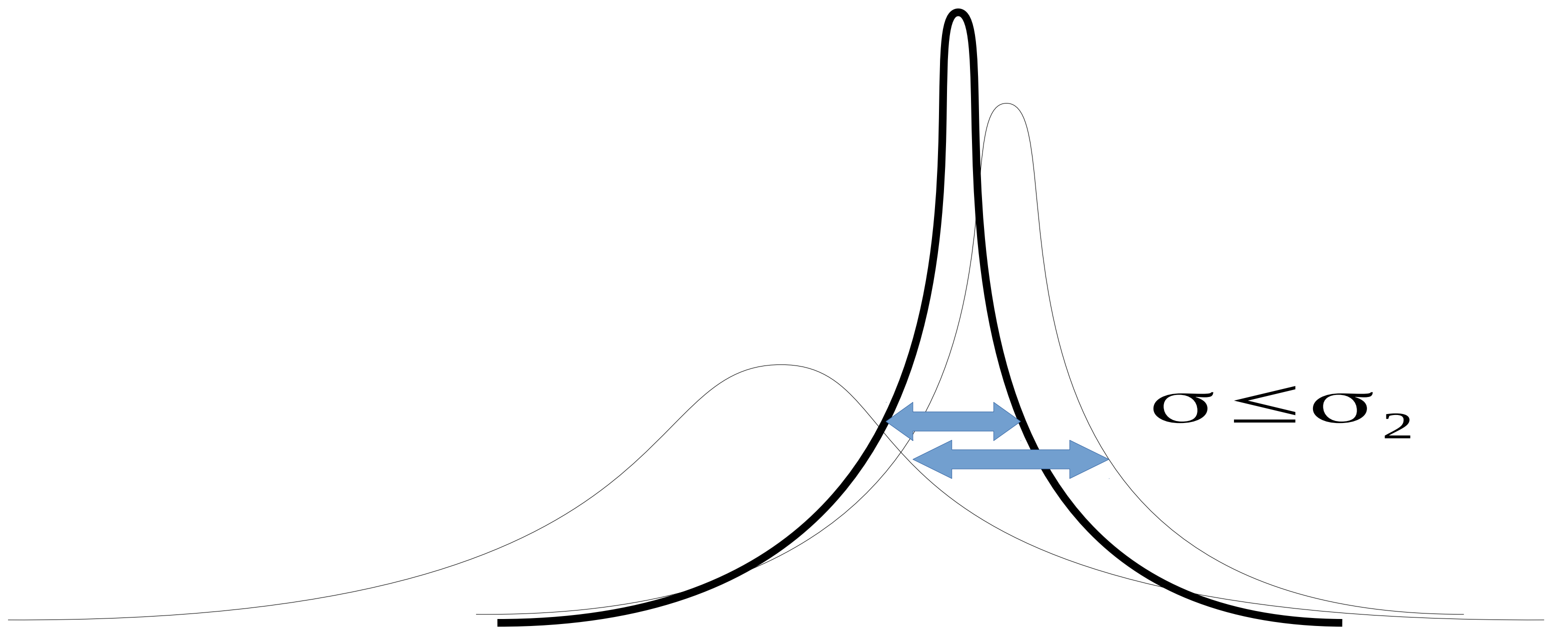
Precision (= inverse of Variance) is an **Additive value** !!!

Precision Increase !

$$\frac{1}{\sigma^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \quad \textbf{=>} \quad \frac{1}{\sigma^2} \geq \frac{1}{\sigma_1^2} \quad \textbf{AND} \quad \frac{1}{\sigma^2} \geq \frac{1}{\sigma_2^2}$$

$$\textbf{=>} \quad \sigma \leq \sigma_1 \quad \textbf{AND} \quad \sigma \leq \sigma_2$$

Both Y1 precision is increased even when merging innacurate Y2
AND Y2 precision is increased even when merging innacurate Y1



StandardDeviation compare with Average Y1...YN / Law

Remarks: $\frac{1}{\sigma^2}=\frac{1}{\sigma_1^2}+\frac{1}{\sigma_2^2}$

If $\sigma_1=\sigma_2=\sigma_i$ **then** $\frac{1}{\sigma^2}=\frac{1}{\sigma_i^2}+\frac{1}{\sigma_i^2}=\frac{2}{\sigma_i^2}$

so $\sigma=\frac{\sigma_i}{\sqrt{2}}$

Remember... Big Numbers Law (with N=2!) : Average(Y1...YN) → $N(\mu,\frac{\sigma}{\sqrt{N}})$

Next Merge (avg Y1..2) + Y3

$Y_{1,}\sigma$ $\quad$ $Y_{2,}\sigma$ $\quad$ $Y_{3,}\sigma$

$$\mu:\frac{Y_1+Y_2}{2}$$

$$\sigma:\frac{\sigma}{\sqrt{2}}$$

$$\mu:\frac{2}{3}\left(\frac{Y_1+Y_2}{2}\right)+\frac{1}{3}(Y3)=\frac{1}{3}(Y_1+Y_2+Y_3)$$

$$\frac{1}{\sigma^2}:\frac{1}{\left(\frac{\sigma}{\sqrt{2}}\right)^2}+\frac{1}{\sigma^2}=\frac{3}{\sigma^2}=\frac{1}{\left(\frac{\sigma}{\sqrt{3}}\right)^2}$$

Compare Avg / Repeat Merge: Y1 → merge Y2 → merge Y3 → .. merge Yn

THE optimal estimation for Y1,Y2, … Yn is : $Average(Y_{1..n}) = \frac{Y_1 + Y_2 + \ldots Y_n}{n}$

Avg Dev~ $\frac{\sigma}{\sqrt{N}}$

Compare with "sub(?)-optimal" **merging 1 by 1**
only remembering last avg and last stddev ...

Y_1, σ → Y_2, σ → Y_3, σ → … → Y_n, σ

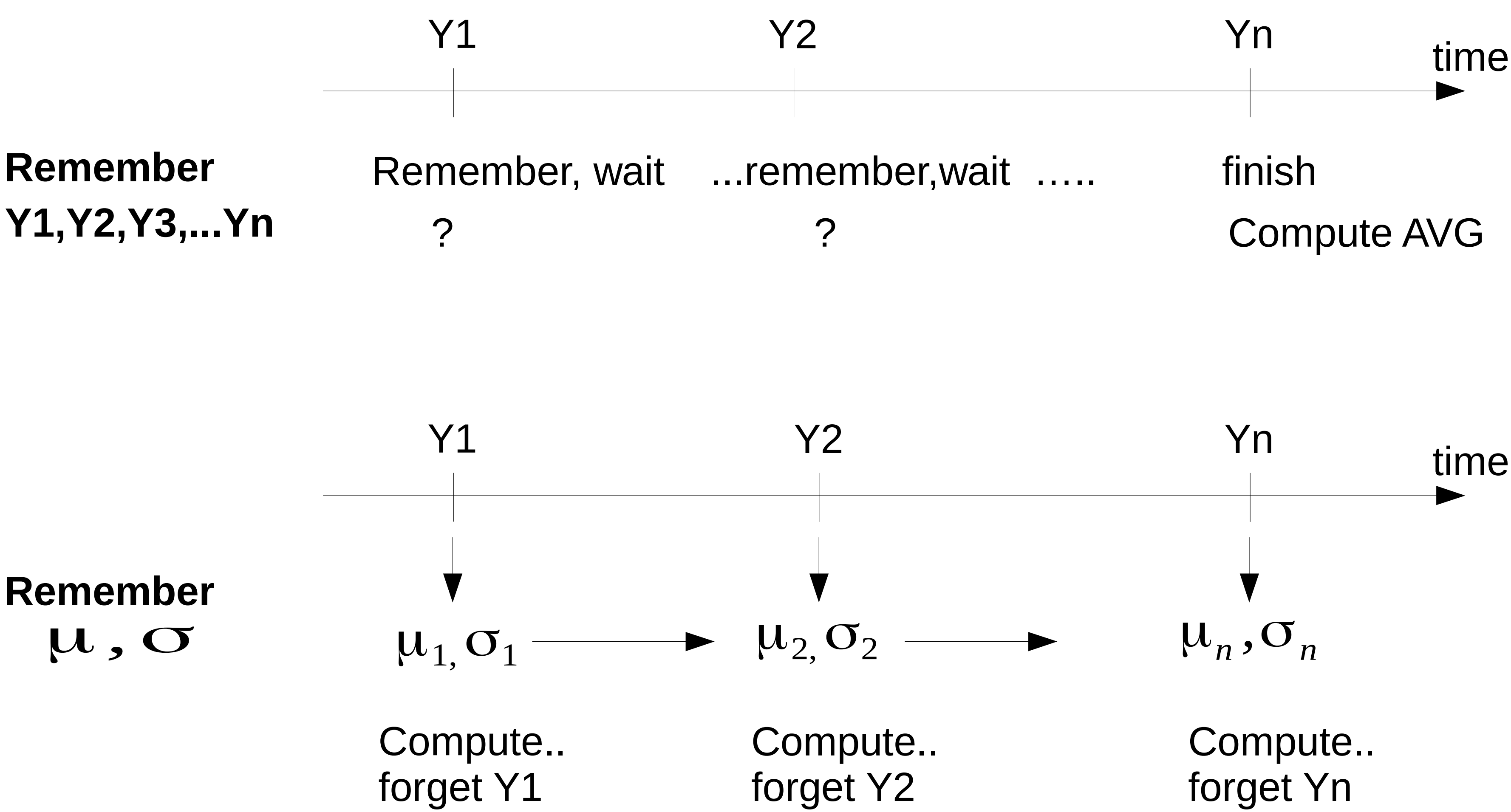
$\frac{Y_1 + Y_2}{2}$, $\frac{\sigma}{\sqrt{2}}$

$\frac{Y_1 + Y_2 + Y_3}{3}$, $\frac{\sigma}{\sqrt{3}}$

$\frac{Y_1 + Y_2 + \ldots + Y_n}{n}$, $\frac{\sigma}{\sqrt{n}}$

SAME Optimal result !!
All-in-One
OR
One-by-One

Kalman = THE Optimal Estimator with low memory Filter requirement (higher CPU than final avg)



Re-interpret StdDev with K ratio

$$\sigma^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$

$$K = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}$$

$$\Rightarrow \quad \sigma^2 = K \sigma_1^2 = (1 - K) \sigma_2^2$$

result close to Y1 with weigth K … Y1 variance is shrink by K
result close to Y2 with weigth (1-K) … Y2 variance is shrink by 1-K

Y1=Estimation, Y2=Measure …
K=preserve / 1-K=innovation

Y2
= sensor measure
on physical system

Y1
= a-priori estimation
(~ estimation measure
from software system)
Notation: $\widehat{X}^{-}$

Y
= a-posteriori estimation
(updated estimation
for software system)
Notation : $\widehat{X}^{+}$

K = weight ratio to preserve knowledge from a-priori estimation
… 1-K = correction factor with new measure
K = multiplication factor for new variance

Back To Dimension N

X $\begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix}$

$$E(X) = E\begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} E(x_1) \\ E(x_2) \\ \dots \\ E(x_n) \end{pmatrix}$$

$$Covar\,X, Y = (cov(X_i, Y_j))_{ij} = \begin{pmatrix} E(x_{1,} y_1) & E(x_{1,} y_2) & .. & E(x_{1,} y_n) \\ E(x_{2,} y_1) & E(x_{2,} y_2) & .. & E(x_{2,} y_n) \\ \vdots & & & \\ E(x_n, y_1) & E(x_n, y_2) & .. & E(x_n, y_n) \end{pmatrix}$$

$$P = var\,X = (cov\,X_i, X_j)_{ij}$$

Time Update : E(Xn+1|Xn) =? f(Xn)

Given State Model Equation:

$$X_{n+1} = A\,X_n + B\,Un + D\,W_n$$

Then

$$\widehat{X_{n+1}} = E\left(X_{n+1} \mid X_n\right)$$

$$= E\left(A\,X_n + B\,Un + D\,W_n \mid X_n\right)$$

$$= A\,E\left(X_n \mid X_n\right) + B\,E\left(U_n \mid X_n\right) + E\left(D\,W_n \mid X_n\right)$$

$$= A\,\widehat{X}_n + B\,U_n + 0$$

Time Update Pn+1 =? f(Pn)

$$P_{n+1} = Cov\, X_{n+1}$$

$$= E\left(X_{n+1} . {}^{t}X_{n+1}\right)$$

$$= E\left(\left(A X_n + B U_n + D W_n\right) . {}^{t}\left(A X_n + B U_n + D W_n\right)\right)$$

$$= E\left(A X_n {}^{t}X_n {}^{t}A\right) + .. E\left(X_n {}^{t}U_n\right) + .. E\left({}^{t}U_n X_n\right)$$
$$+ E\left(W_n {}^{t}\left(.. X_n .. U_n\right)\right) + E\left(\left(.. X_n .. U_n\right) {}^{t}W_n\right)$$
$$+ E\left(D W_n {}^{t}W_n {}^{t}D\right)$$

$$= A P_n {}^{t}A + D {}^{t}D \qquad + 2 A X_n {}^{t}U_n {}^{t}B$$

Interpretation of Cov Matrix Increase

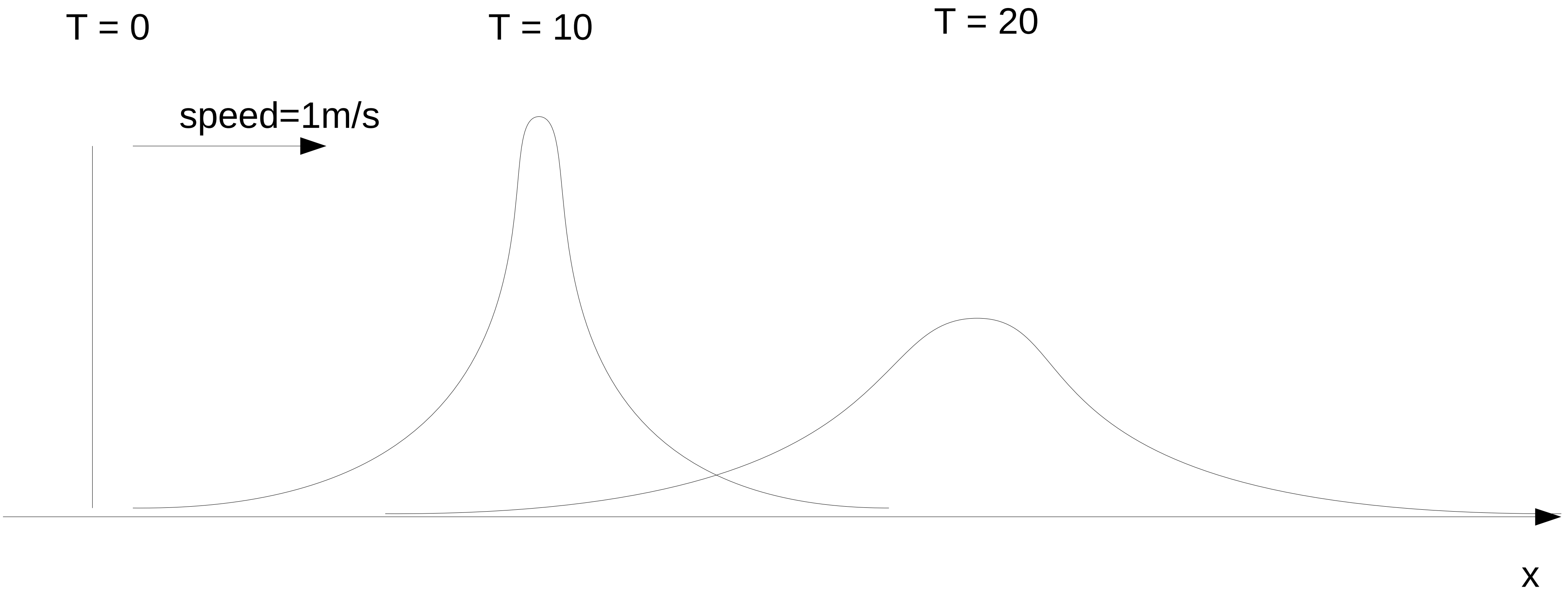
$A P_n{}^t A$ Intrinsinc model variance increase

$D^t D = Q$ Notation: Q covariance of Brownian Noise
= power of noise

$2 A X_n{}^t U_n{}^t B$ variance from model-control

Example of Variance Increase

Walk at ~1m/second during 20 seconds, with CLOSED EYES
Estimate your position & accuracy



Example (2/3)

$$m\ddot{X} = \sum f \pm e \qquad f = 0 \rightarrow \ddot{X} = 0 \pm e\,,\ \dot{X} = cst \pm ..$$

Time Update equation:

$$X_{t+\delta t} = X_t + \dot{X}\,\delta t$$

If Speed is imprecise
=> then next position computation is imprecise

$$X_{t+\delta t} = \left(X_t \pm \sigma_X\right) + \left(\dot{X} \pm \sigma_{\dot{X}}\right)\delta t$$
$$= \left(X_t + \dot{X}\,\delta t\right) + \left(\pm\sigma_X \pm \sigma_{\dot{X}}\,\delta t\right)$$

$$\sigma_{X\,t+dt} : merge \pm\sigma_X\,, \pm\sigma_{\dot{X}}\,\delta t$$

Example (3/3)

$$X = \begin{pmatrix} x \\ \dot{x} \end{pmatrix} \qquad \dot{X} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} X + \begin{pmatrix} 0 \\ f/m \end{pmatrix}$$

$$A P^t A = \begin{pmatrix} \sigma^2_{xx} & \sigma^2_{x\dot{x}} \\ .. & \sigma^2_{\dot{x}\dot{x}} \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \sigma^2_{xx} & \sigma^2_{x\dot{x}} \\ .. & \sigma^2_{\dot{x}\dot{x}} \end{pmatrix}$$

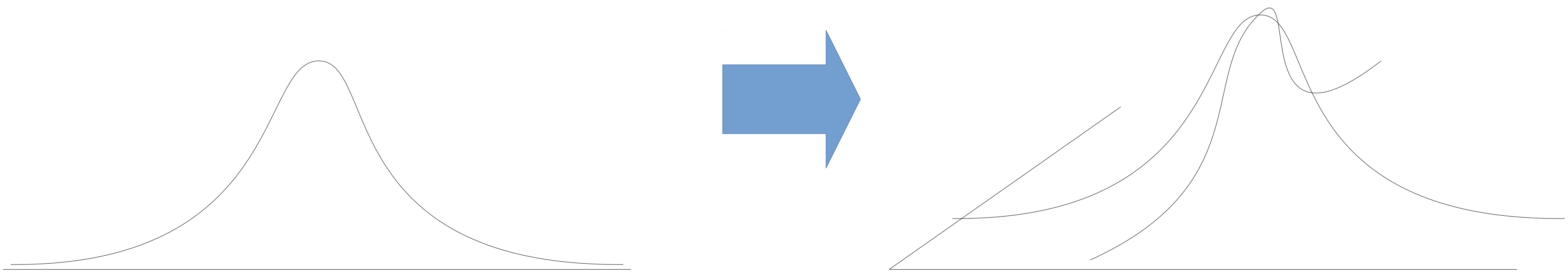
$$= \begin{pmatrix} 0 & \sigma^2_{xx} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \sigma^2_{xx} & \sigma^2_{x\dot{x}} \\ .. & \sigma^2_{\dot{x}\dot{x}} \end{pmatrix}$$

$$= \begin{pmatrix} \sigma^2_{xx}\sigma^2_{\dot{x}x} & \sigma^2_{xx}\sigma^2_{\dot{x}\dot{x}} \\ 0 & 0 \end{pmatrix}$$

Probability Distribution in Dim N

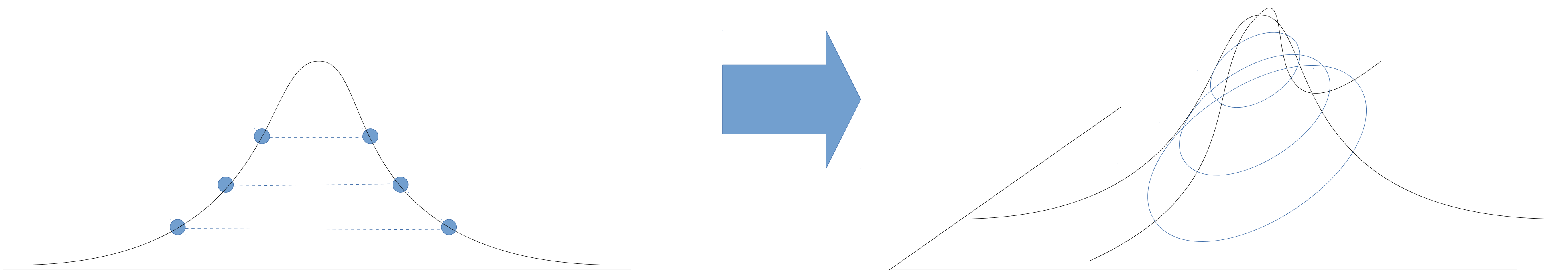
Dimension 1

Dimension 2,3,..N



Dim N : Gaussian - Ellipsoids …

Dimension 1 Dimension 2,3,..N

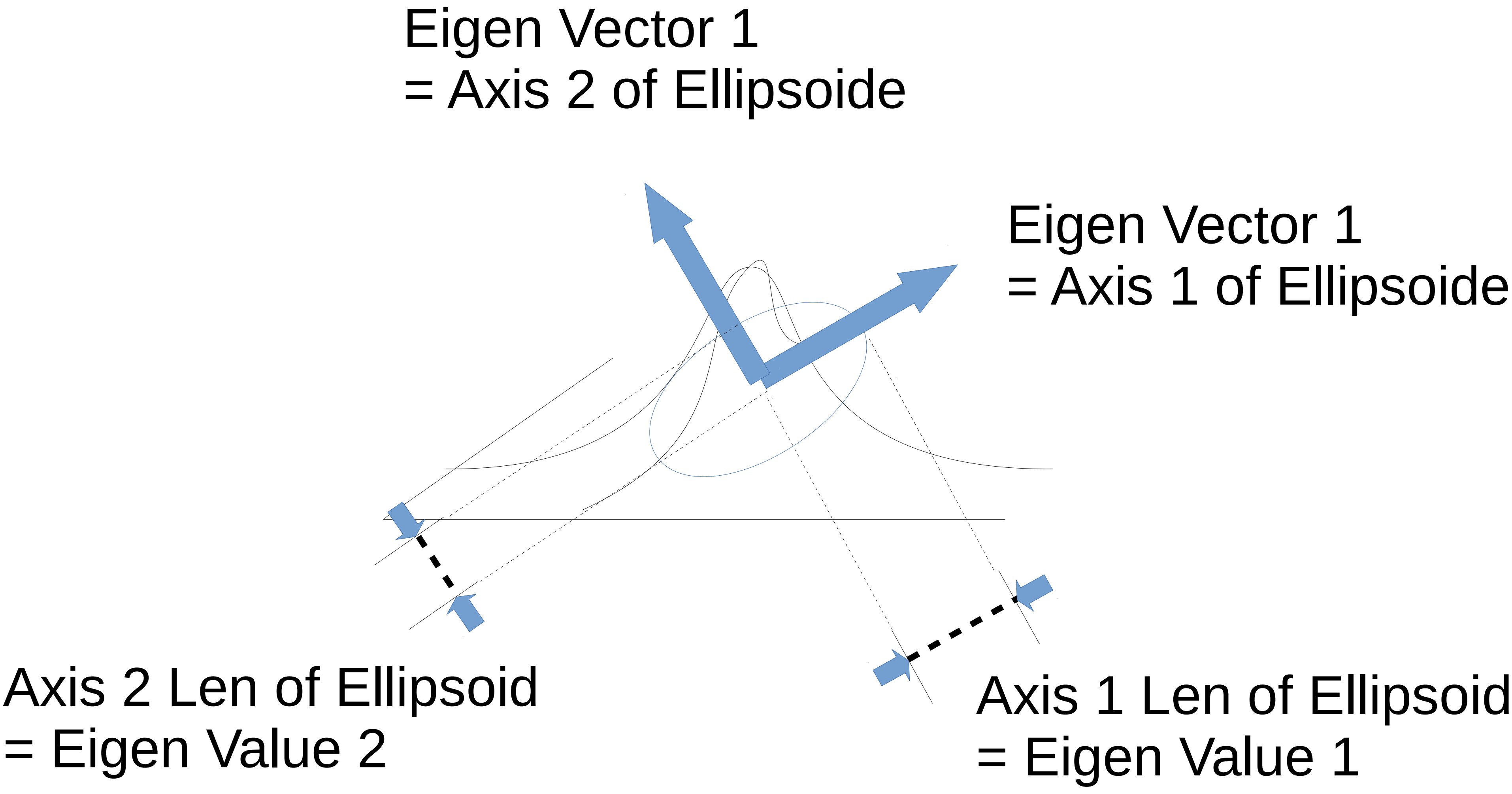


Equi-probability curves = Elipsoid surfaces

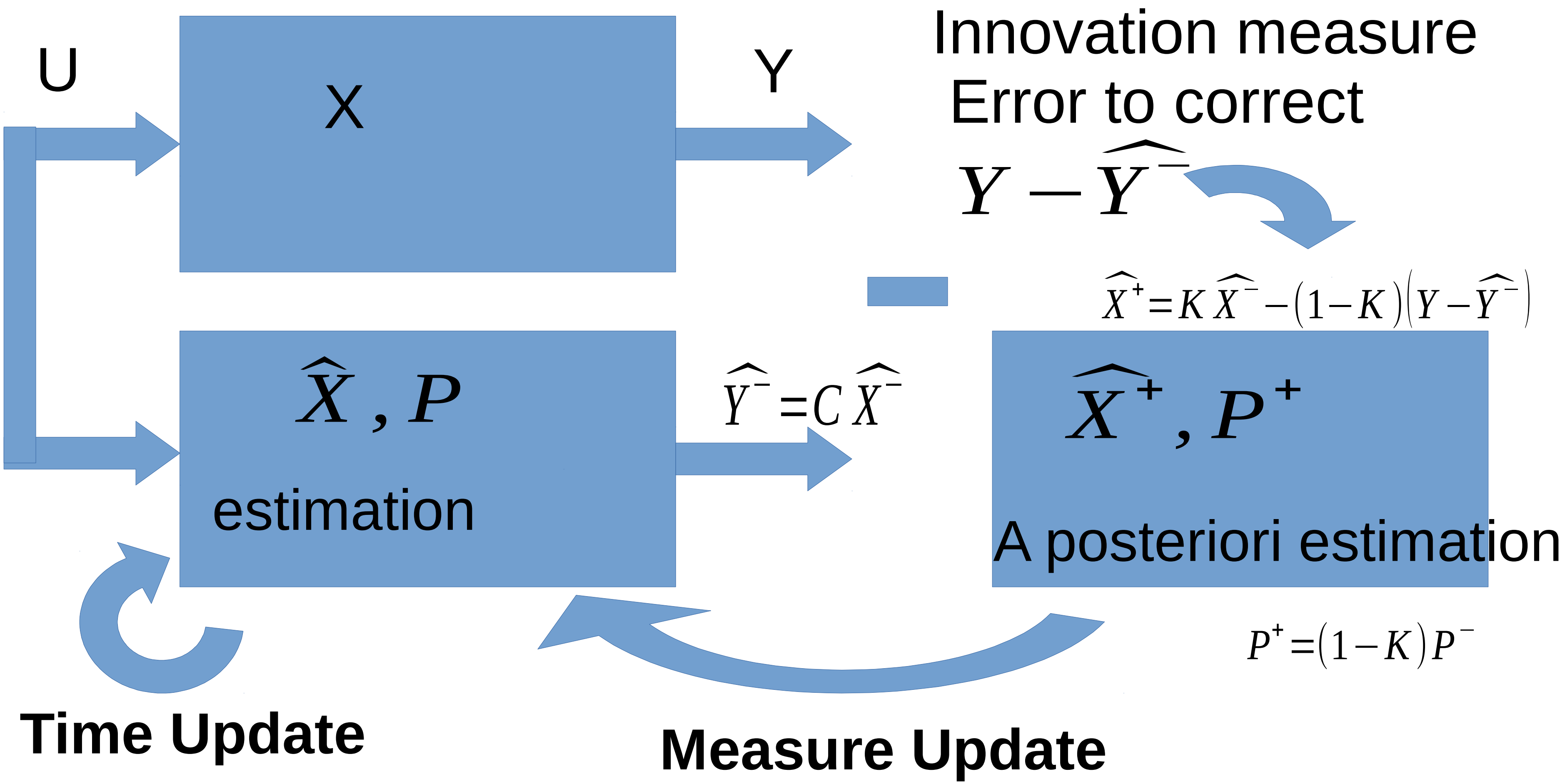
Covariance Matric is a Symmetric >=0 matrix
.. is a scalar product
.. in a changement of axis, is a normal scalar product matrix
.. is Diagonalisable in a changement of axis (R)

$$P = R^{-1} D R$$

Ellipsis Axis : Eigen Vector – Value



Kalman Observe System Drawing



K = kalman gain = weight ratio predict vs measure

Kalman Equations

$$\mathbf{x}_k = \mathbf{F}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k \qquad \mathrm{Q}_k\!: \mathbf{w}_k \sim \mathcal{N}(0, \mathbf{Q}_k).$$

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \qquad \mathrm{R}_k\!: \mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$$

WIKIPEDIA
The Free Encyclopedia

Predict [edit]

Predicted (*a priori*) state estimate	$\hat{\mathbf{x}}_{k\mid k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1\mid k-1} + \mathbf{B}_k \mathbf{u}_k$
Predicted (*a priori*) estimate covariance	$\mathbf{P}_{k\mid k-1} = \mathbf{F}_k \mathbf{P}_{k-1\mid k-1} \mathbf{F}_k^{\mathrm{T}} + \mathbf{Q}_k$

Update [edit]

Innovation or measurement residual	$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k\mid k-1}$
Innovation (or residual) covariance	$\mathbf{S}_k = \mathbf{R}_k + \mathbf{H}_k \mathbf{P}_{k\mid k-1} \mathbf{H}_k^{\mathrm{T}}$
Optimal Kalman gain	$\mathbf{K}_k = \mathbf{P}_{k\mid k-1} \mathbf{H}_k^{\mathrm{T}} \mathbf{S}_k^{-1}$
Updated (*a posteriori*) state estimate	$\hat{\mathbf{x}}_{k\mid k} = \hat{\mathbf{x}}_{k\mid k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k$
Updated (*a posteriori*) estimate covariance	$\mathbf{P}_{k\mid k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k\mid k-1}$

Kalman Equations (rewrite notations)

Model

$$\begin{cases} X_{n+1} = A\,Xn + B\,U_n + D\,W_n \\ Y_n = C\,X_n + E\,W_n \end{cases}$$

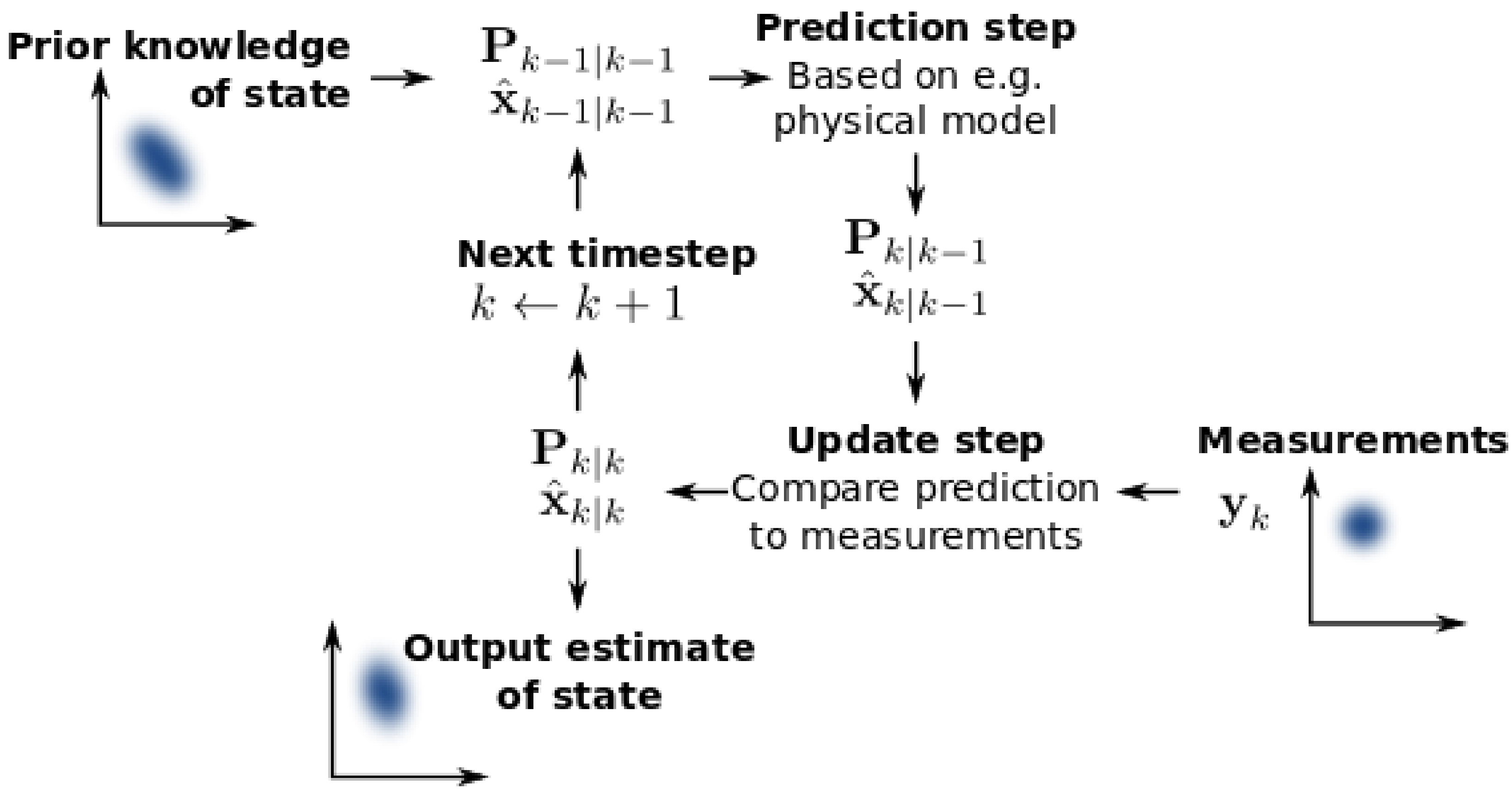
Time Update

$$\begin{cases} \widehat{X_{n+1}^{-}} = A\,\widehat{X_n^{-}} + B\,Un \\ P_{n+1} = A\,P_n{}^t A + D^t D \end{cases}$$

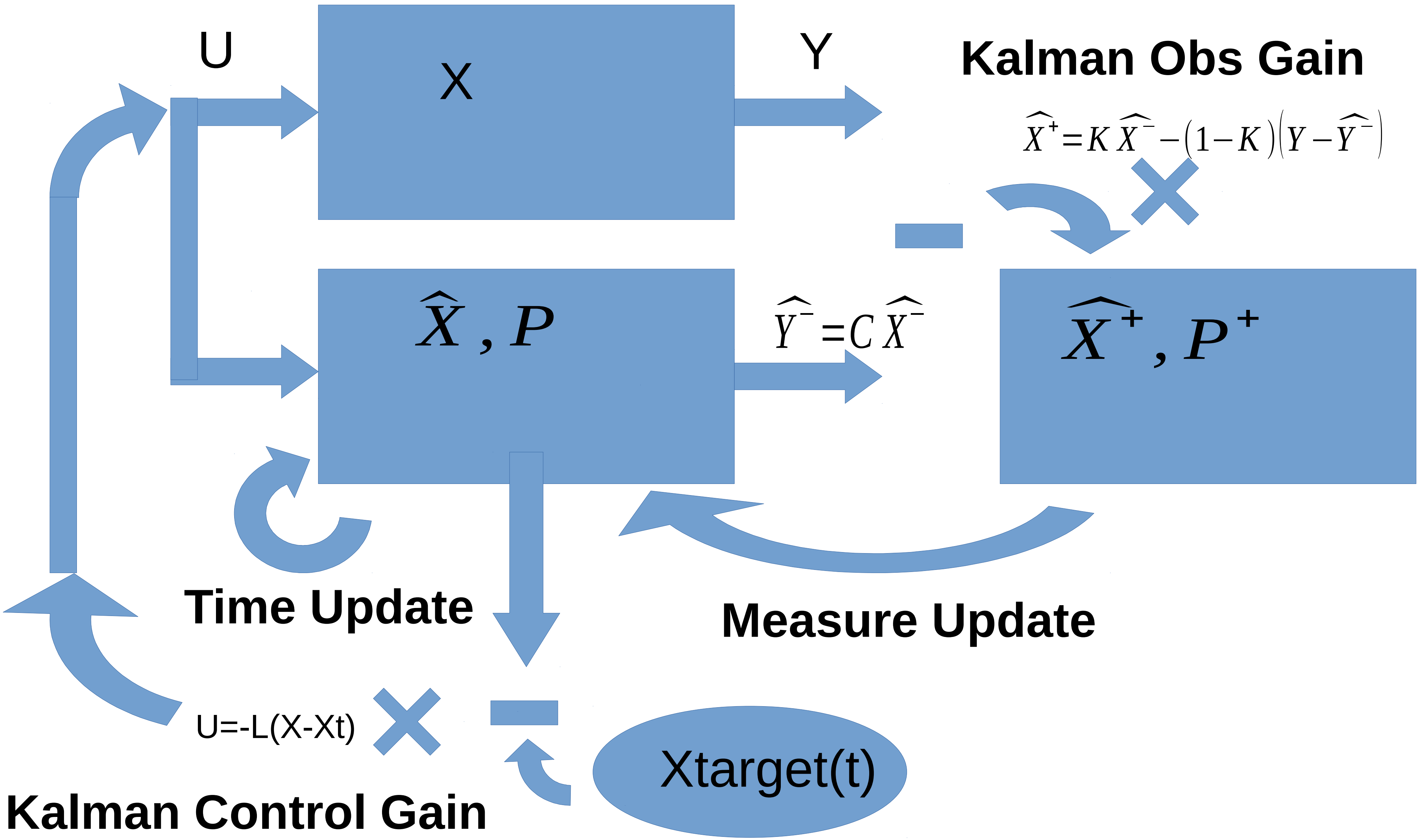
Measure Update

$$\begin{cases} \widehat{X_{n+1}^{+}} = \widehat{X_n^{-}} + K\left(Y_n - C\,\widehat{X_n^{-}}\right) \\ P_{n+1} = \left(I - K\,C\right) P_n \end{cases}$$

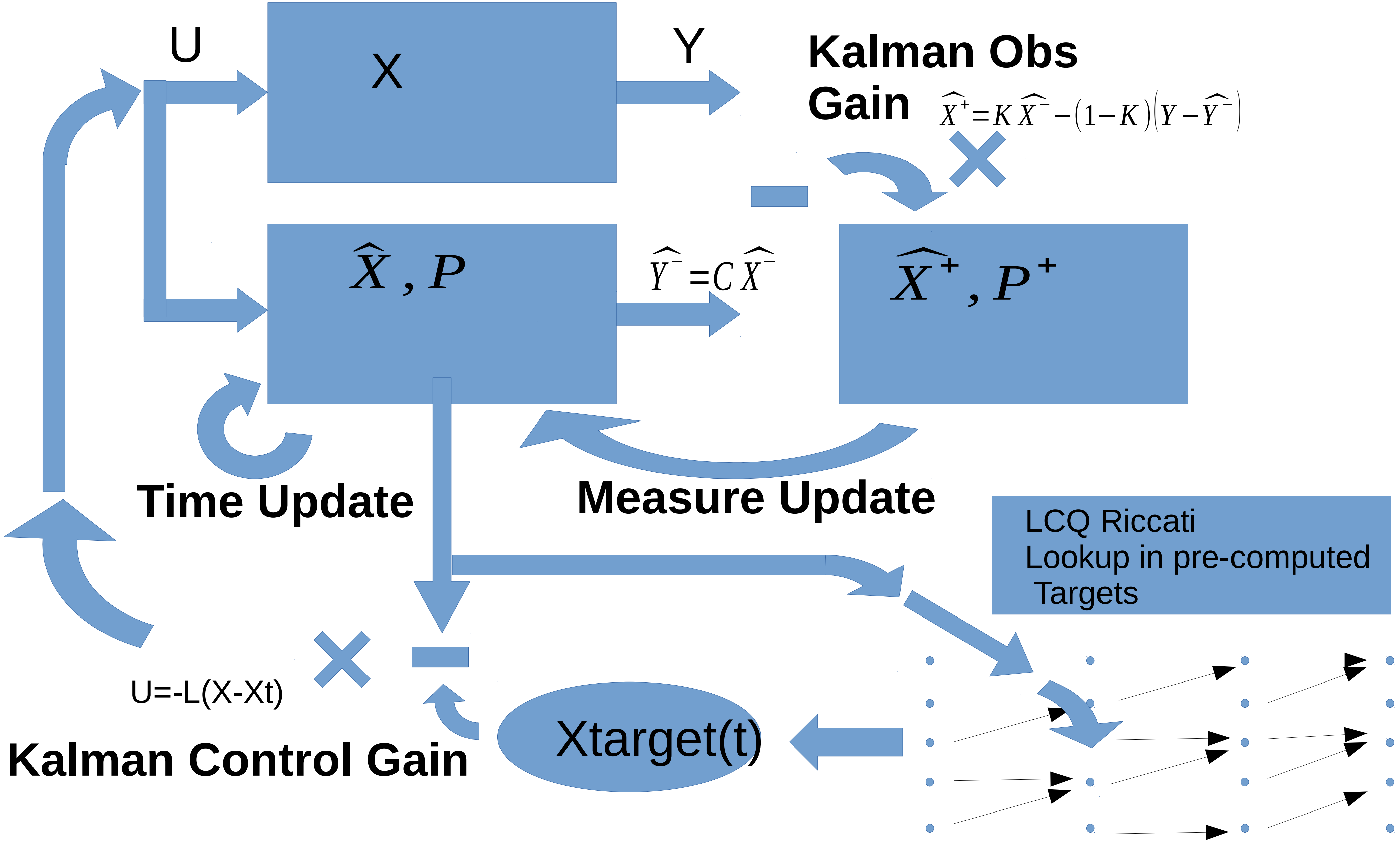
Kalman System Drawing



Kalman Control + Observe System Drawing



Kalman Control + LCQ + Observe



More Kalman Filters...

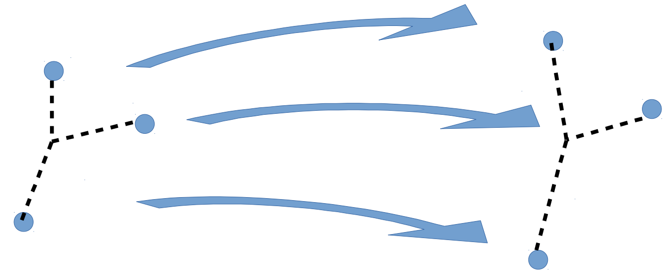
Kalman filter theory … assumption = Linear System
In ~ 1958 => improvements for Non-linear System since...

Extended Kalman Filter : linearised

$$X_{n+1} = f(X_n, U_n, W_n)$$

$$A_{ij\,n} = \frac{\partial f}{\partial ij}(X_n, U_n, W_n)$$

Unscented Kalman Filter : linearised by “parts”



Auto-Adaptative Parameters (Learning Reinforcement)

Suppose A is not exactly known … only estimated
=> incorporate A param(s) as X params

Linearise $A = A^0 + \delta A \qquad X = X^0 + \delta X$

$$A X = (A^0 + \delta A)(X^0 + \delta X)$$
$$= A^0 X^0 + A^0 \delta X + \delta A X^0 + ..$$

Use augmented system:

$$\mathcal{X} = \begin{pmatrix} \delta X \\ \delta A \end{pmatrix} \qquad \dot{\mathcal{X}} = \begin{pmatrix} \dot{\delta X} \\ \dot{\delta A} \end{pmatrix} = \begin{pmatrix} \dot{(A X)} + B U + D E \\ 0 \end{pmatrix} = \ldots$$

Sample Applications

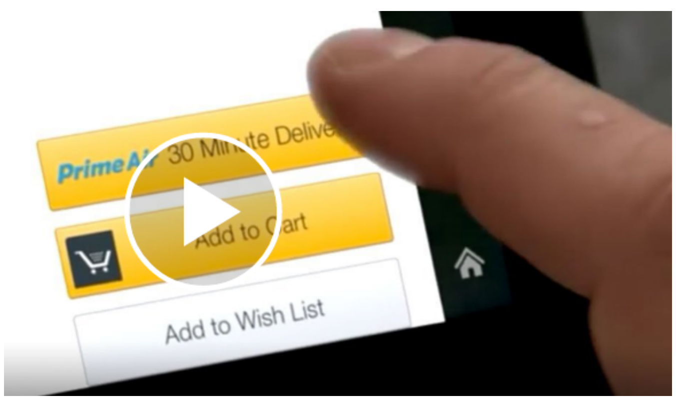
Drone with 6 Propelers (> 4)

… still stable after 1 broken propeler

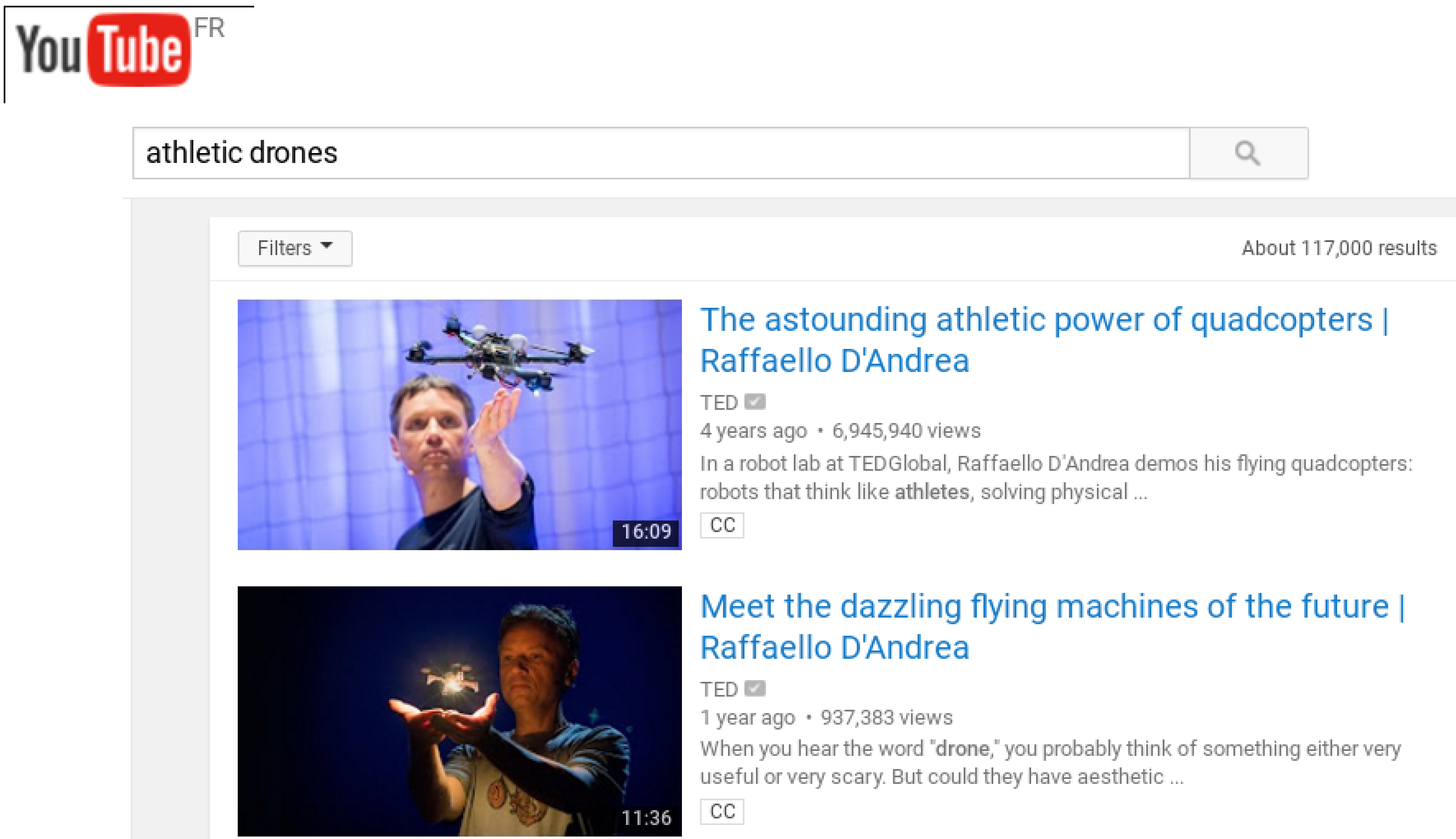
… BECAUSE of auto re-calibration learning

… still partially controllable (to land) after 2 broken Propelers

Could deliver products safely with Drones over public areas ...



Sample TED Video



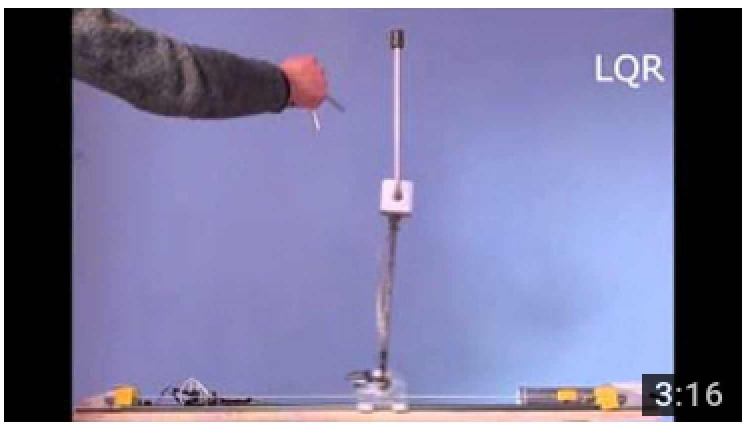
Sample Video

YouTube FR

double inverted pendulum

Filters

About 8,640 results



Control of Double Inverted Pendulum, WETI Gdańsk

Maks K

3 years ago • 64,442 views

Authors: Maksymilian Kunt, Adrian Szwaba.

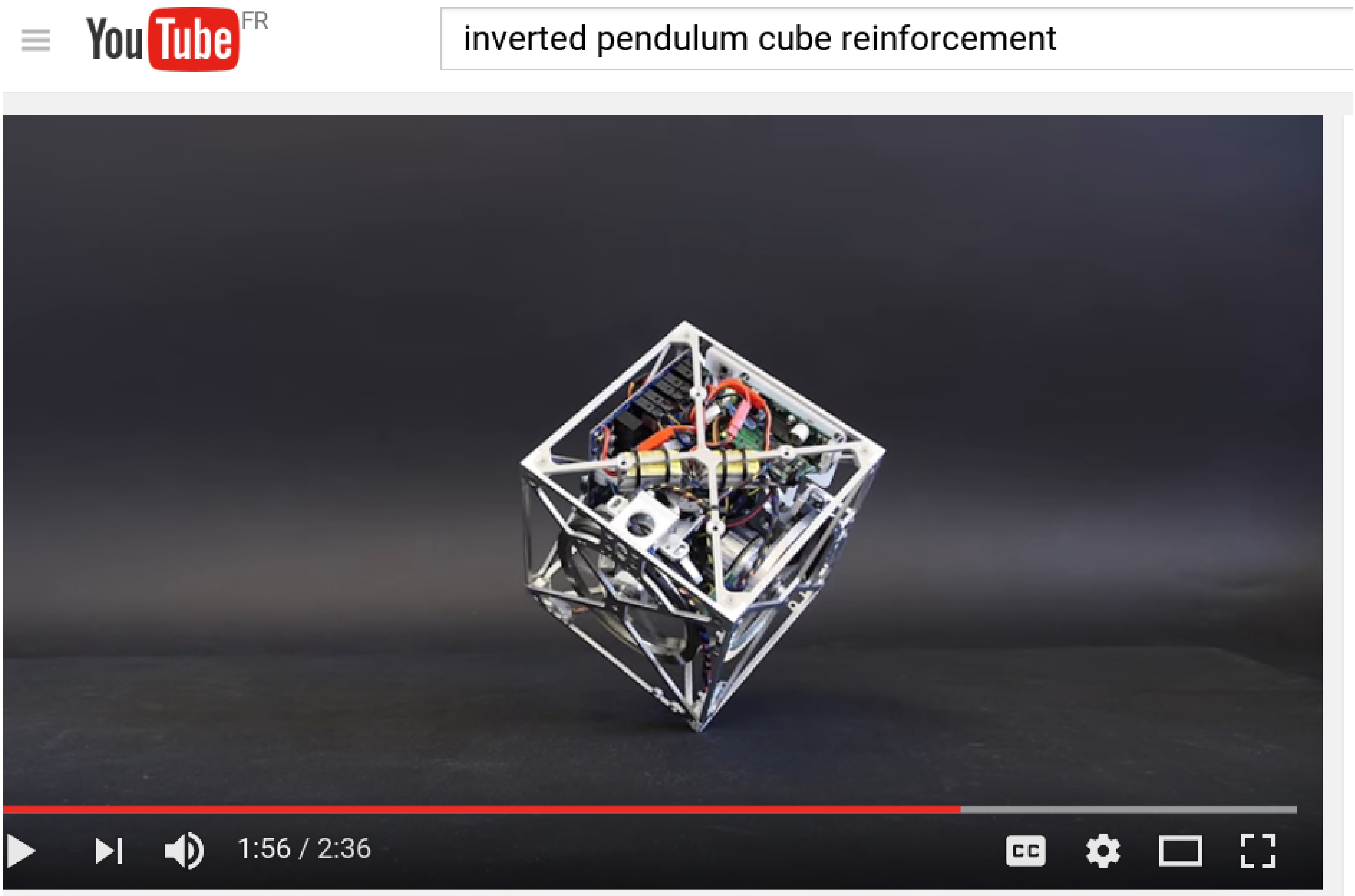
Double Pendulum on a Cart

Tobias Glück

6 years ago • 117,212 views

Double Pendulum on a Cart 12 x point-to-point control and 4 x side-stepping Two-degrees-of-freedom design: Constrained ...

Sample Video



Conclusion

Kalman Filter is Awesome