Kalman Filter

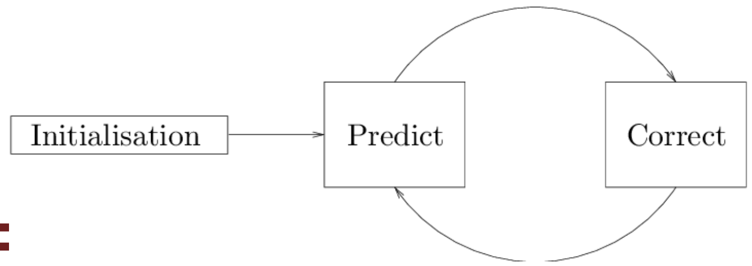
Nacho Sañudo
University of Modena and Reggio Emilia
Ignacio.sanudoolmedo@unimore.it

Credits to Udacity self-driving course

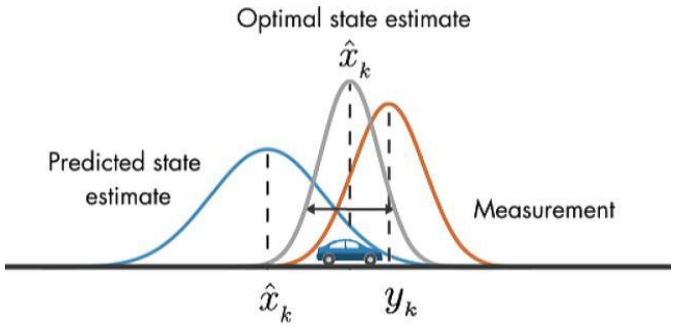
Introduction

- Let’s suppose that we would like to estimate the velocity of the car
- Available info.
  - T=0 -> 0 meters
  - T=1 -> 25 meters
  - T=2 -> 50 meters
  - T=3 -> 75 meters
  - T=4 -> 100 meters
- How fast is the car moving?

Example Kalman Filter

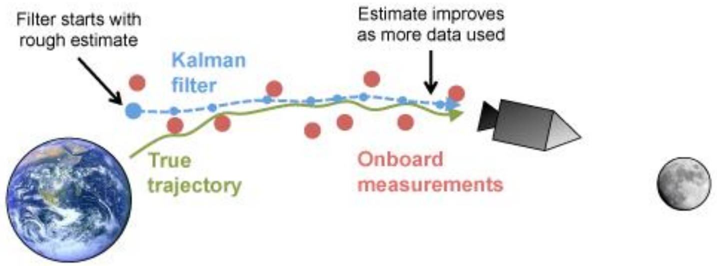


- We start by reading the output of the GPS
- We can use a probability distribution to characterize the position of the car (**belief**)
- The **dynamic model** of the vehicle is then use to prodict how the car moves in the environment
  - For instance: location = previous location + velocity * δt
- Then, we get again a new **measurement** and we produce a new probability distribution by merging both PDFs to produce a final prediction

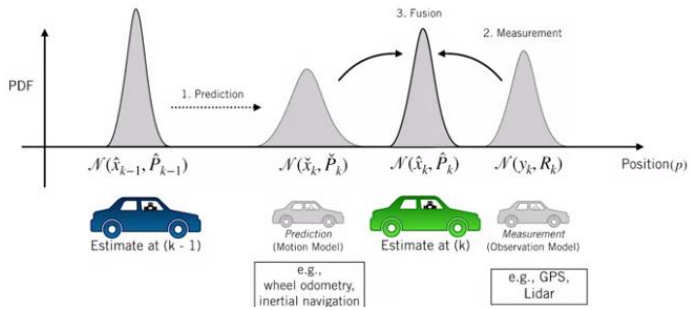


Kalman Filter

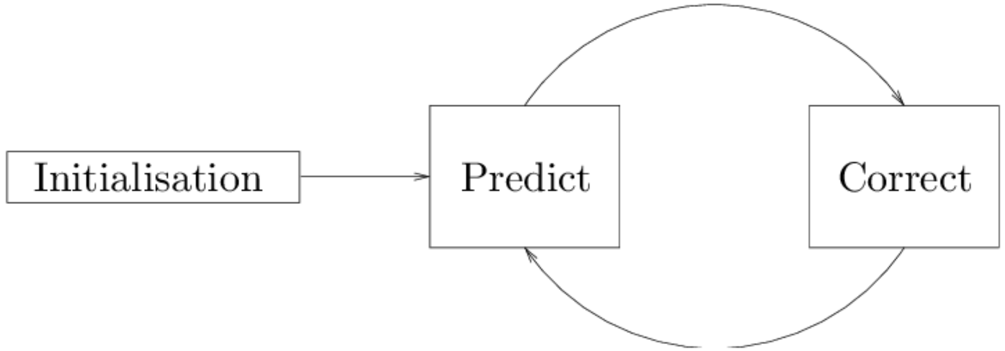
- Created in 1960 by Rudolf E. Kalman
  - Used in the trajectory estimation system for the Apollo program
- **Kalman filter is a technique for estimating the state of a system**
  - Mathematical process that uses equations and data (from the sensors for instance) to estimate a value (e.g., **position or velocity**) of the objects
  - It is used to **correct (sensors are not perfect) and/or predict** the values of the objects under analysis



Kalman Filter



› The goal of the Kalman filter is to get a probabilistic estimate of the vehicles' state and update it in real time using two steps; **prediction and correction (or update)**
  - **Prediction** step uses the motion model which defines how the vehicle evolves over time (IMU, car's information…)
  - In the **update** state we use the information of the GPS/radar/LiDAR to correct the previous prediction through the measurement model
    - › New measurements gives us information with a small variance

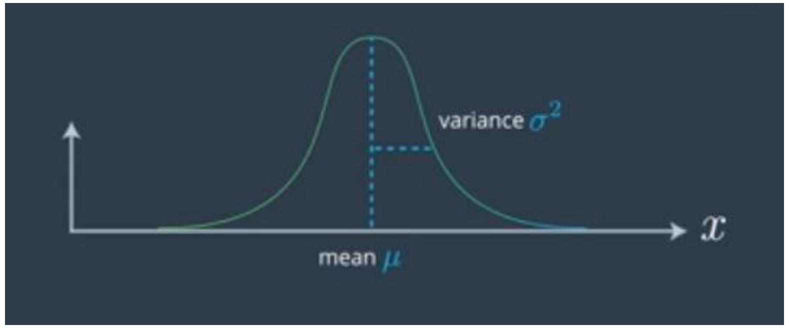


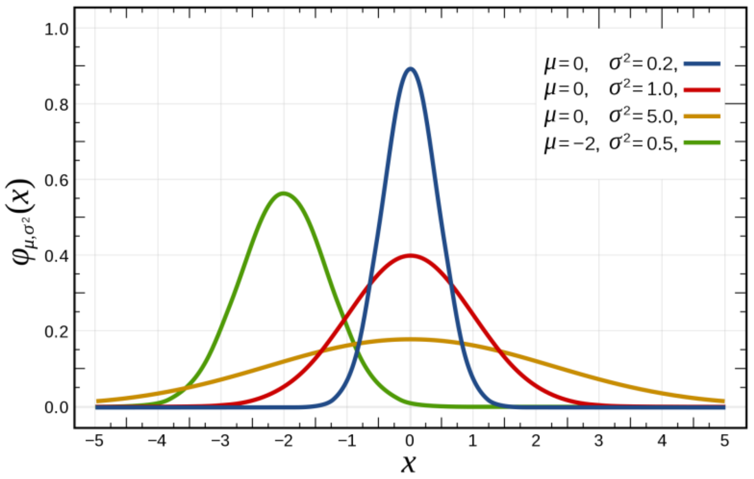
Gaussians

- **In Kalman Filters the distribution is given by a Gaussian**
- A Gaussian (or normal) distribution is a type of continuous probability distribution for a real-valued random variable
  - The area underneath sums up to 1
  - The parameter **μ** is the mean or expectation of the distribution (e.g., expectation of the position)
  - The variance of the distribution is $\boldsymbol{\sigma^2}$
    - **a measure of certainty**
      - The larger the more uncertain we are
- 1-D Gaussian (μ, σ^2)
  - **Objective**: maintain a μ and σ^2 as our best estimate of the location/value of the object we are trying to find

Gaussian function formula

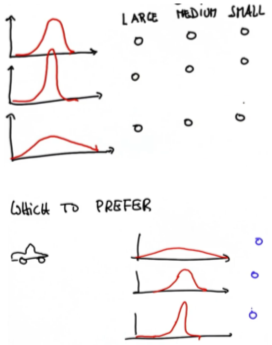
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$



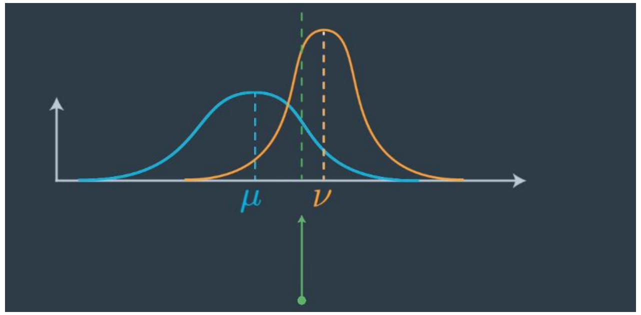


Kalman Filter

Covariance?



- Let’s suppose that we are computing the position of the vehicle which is represented using a Gaussian distribution
  - New measurements gives us information with a small variance
  - **Combining** the measurements with the **prior** distribution we can improve the estimation (**posterior**) of the localization
- If we have **μ estimation of the position and *v* measurement from the sensors. How is the new Gaussian?**



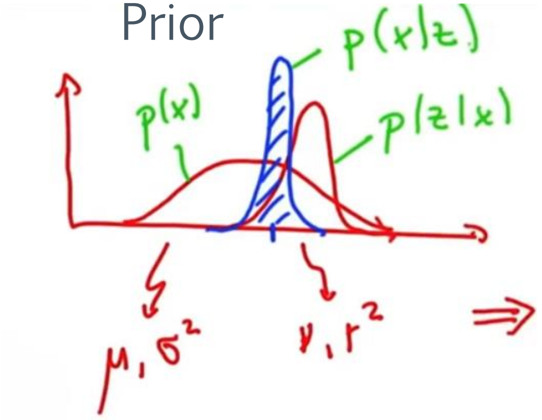
μ is the **prior** mean and ν is the new posterior measurement

Correct/Update

- We need to use the information provided by the **measurement model** to **update** the **prior estimation**
  - We **multiply two Gaussians as in the Bayes Rule (we have to compute *μ'*, σ²)**
    - the new mean, ***μ'***, is the weighted sum of the old means
    - **the prior Gaussian has a higher uncertainty**
      - **Sigma square (σ²) is larger, so that Nu(*v*) has a higher weight than μ. So, the resultant mean will be closer to *v* than the μ**
        - **<u>We gain information</u>**

Posterior

Prior



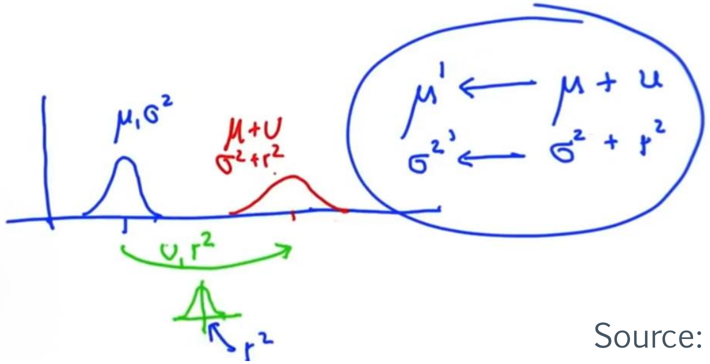
Measurement probability

$$\mu' = \frac{r^2\mu + \sigma^2 v}{r^2 + \sigma^2}$$

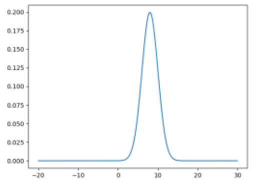
$$\sigma^{2'} = \frac{1}{\frac{1}{r^2} + \frac{1}{\sigma^2}}$$

Prediction

- The **prediction step** produces estimates of the current variables **considering the uncertainties**
  - The estimates are based on a model that "models" **how the system changes over time**
  - In this case we apply the **motion model** (for instance, the acceleration of the car)
  - **By performing just an addition of the Gaussians, i.e., addition of the prior mean (& variance) with the new ones (e.g., new_location=old_location+velocity)**
    - A new mean (***μ’***) is the old mean plus the motion (***u***)
      - if you move over 100 meters, this will be 100 meters + my position
    - The new variance (**σ²’**) is **σ²** plus the variance of the motion Gaussian
      - **<u>We loose information that’s why the variance is higher</u>**

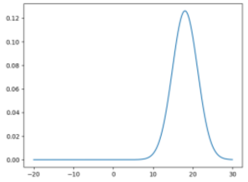


Source: Udacity



$\mu = 8$
$\sigma^2 = 4$

$u = 10$
$r^2 = 4$



$\mu' = ?$
$\sigma^{2\prime} = ?$

Program 1D kalman filter

```
measurements = [3.,4.,6.,7.,8.]
motion = [1., 2., 1., 1., 1.]
measurement_sigma = 4.  #measurement uncertainty
motion_sigma = 2.       #motion uncertainty
mu = 0                  #initial position estimate (and mu later represents the position)
position_sigma = 10000. #position uncertainty
```

position uncertainty

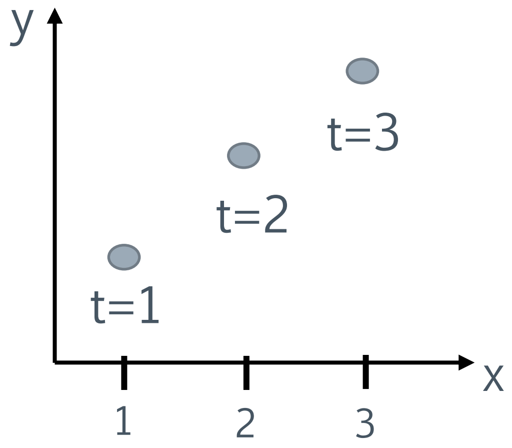
```
update:  [2.998800479808077, 3.9984006397441023]
predict:  [3.998800479808077, 5.998400639744102]
update:  [3.9995201151723596, 2.399744061425258]
predict:  [5.999520115172359, 4.399744061425258]
update:  [5.999771476452553, 2.0951800575117594]
predict:  [6.999771476452553, 4.09518005751176]
update:  [6.999887081672885, 2.0235152416216957]
predict:  [7.999887081672885, 4.023515241621696]
update:  [7.99994370630642, 2.0058615808441944]
predict:  [8.99994370630642, 4.005861580844194]
```

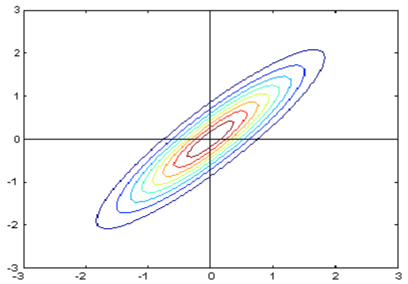
The position_sigma grows with predictions and decreases with the measurements

2D Kalman filter

- Until now we considered the kalman filter in just one dimension
  - **In real world we consider more dimensions**
- Kalman filter is able to extract the velocity of the object, and from there it is possible to perform predictions
  - In the example we observe only positions the velocity is obtained from the measurements
- KF doesn’t need the full story to determine hidden variables, it only uses past and current measurements
- To model this we need high dimensional Gaussians
  - **Multivariate Gaussians**

What is my velocity?



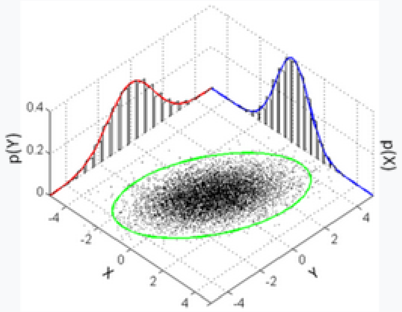


Multivariate Gaussian

- Multivariate Gaussian distribution, is a generalization of the one-dimensional (univariate) normal distribution to higher dimensions
  - The mean (μ) is a vector with one element for each of the dimensions
    - **Prediction model: [position,velocity]**
    - **Sensor model: [position,velocity] from sensors**
  - The covariance (Σ) is a matrix with DxD where D is the dimension
    - **Prediction model: covariance matrix that describes the relationship between variables**
    - **Sensor model: noise of the sensors**

Multivariate normal

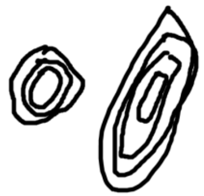
Probability density function



Many sample points from a multivariate normal distribution with $\boldsymbol{\mu} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & 3/5 \\ 3/5 & 2 \end{bmatrix}$, shown along with the 3-sigma ellipse, the two marginal distributions, and the two 1-d histograms.

Notation	$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
Parameters	$\boldsymbol{\mu} \in \mathbf{R}^k$ — location $\boldsymbol{\Sigma} \in \mathbf{R}^{k \times k}$ — covariance (positive semi-definite matrix)
Support	$\boldsymbol{x} \in \boldsymbol{\mu} + \mathrm{span}(\boldsymbol{\Sigma}) \subseteq \mathbf{R}^k$
PDF	$(2\pi)^{-\frac{k}{2}} \det(\boldsymbol{\Sigma})^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\top} \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}$, exists only when $\boldsymbol{\Sigma}$ is positive-definite
Mean	$\boldsymbol{\mu}$
Mode	$\boldsymbol{\mu}$
Variance	$\boldsymbol{\Sigma}$
Entropy	$\frac{1}{2} \ln \det(2\pi e \boldsymbol{\Sigma})$

Multivariate Gaussian

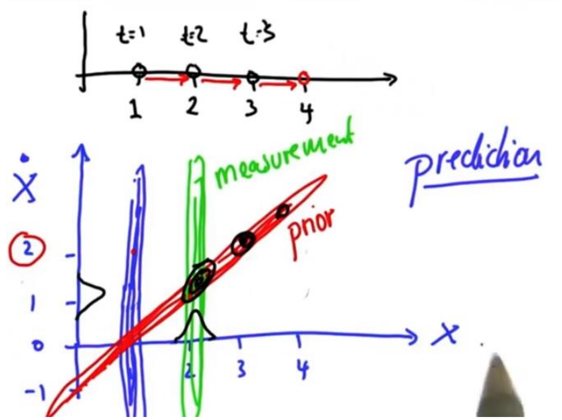


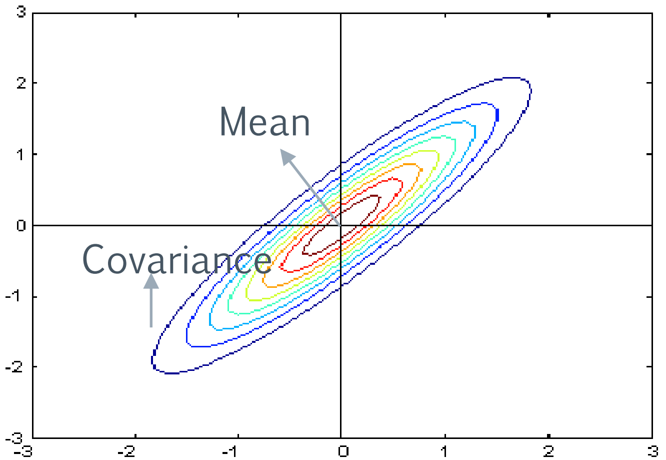
Small and large uncertainties

- 2D estimate (one for the location and the other for the velocity)
  - x -> Position | $\dot{x}$ -> Velocity
  - **If we multiply the prior with the prediction, we get a Gaussian as a good estimate of my position and velocity**
    - We can infer the velocity from this
- We can do predictions even if we are unable to measure velocity correctly
  - KF uses imperfect measurements and produces estimates of unknown variables

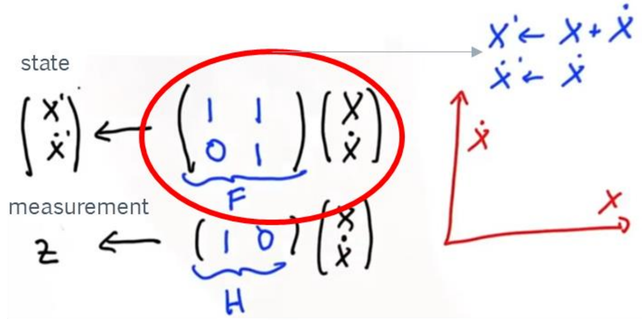
How is the correlation between position and velocity assuming that I'm in x=1?

How is the Gaussian?



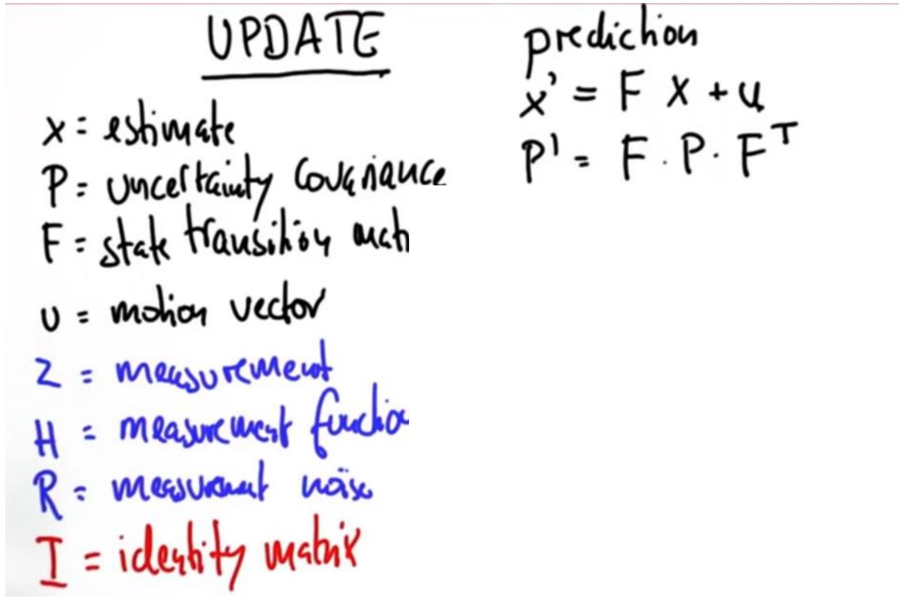


Kalman filter design (Prediction)



Source: Udacity

- **x** is the values of all the variables (or states) you’re considering in your system (e.g., position and velocity)
  - **represents the estimate of the prediction function**
- **P** is the covariance matrix**, holds the current uncertainty of the states**
- **F** is the update matrix; it models how the system evolves over time (state transition matrix)
  - New position→ p’ = p + vΔt
  - New velocity → v’=v
  - Matrix → [(1 ,1),(0,1)]
- **U** is the control matrix that describes how the object changes due to internal or any external force
  - the force of friction of the tires or the force of gravity



Kalman filter design (Update)

- **z** is the value measured from the sensors
- **H** is the matrix that projects the belief of the object's current state into the measurement space of the sensor (measurement function)
- **R** is the noise of the measurement
  - (given by the manufacturer)
- **K** is the Kalman Gain, describes the weight given to the measurements to update the new estimate
  - The error in the estimate and the error in the prediction are assumed
  - The more the error in the prediction the more weight is given to the sensor information (and viceversa)
    - The higher the K the more weight is given to the measurements
- **S** is the error
- **y** is the difference between the measured value and actual value