Laboratorio #10 – Filtro de Kalman

M.C. Fernando Hermosillo Reynoso

fhermosillo@up.edu.mx

Universidad Panamericana

Sesión #10

28 de Enero del 2020

UNIVERSIDAD PANAMERICANA®

- Repositorio GitHub del curso: UP_DSP24
  - Documentos
  - Ejemplos
  - Notas rápidas
  - Laboratorios

Prelab

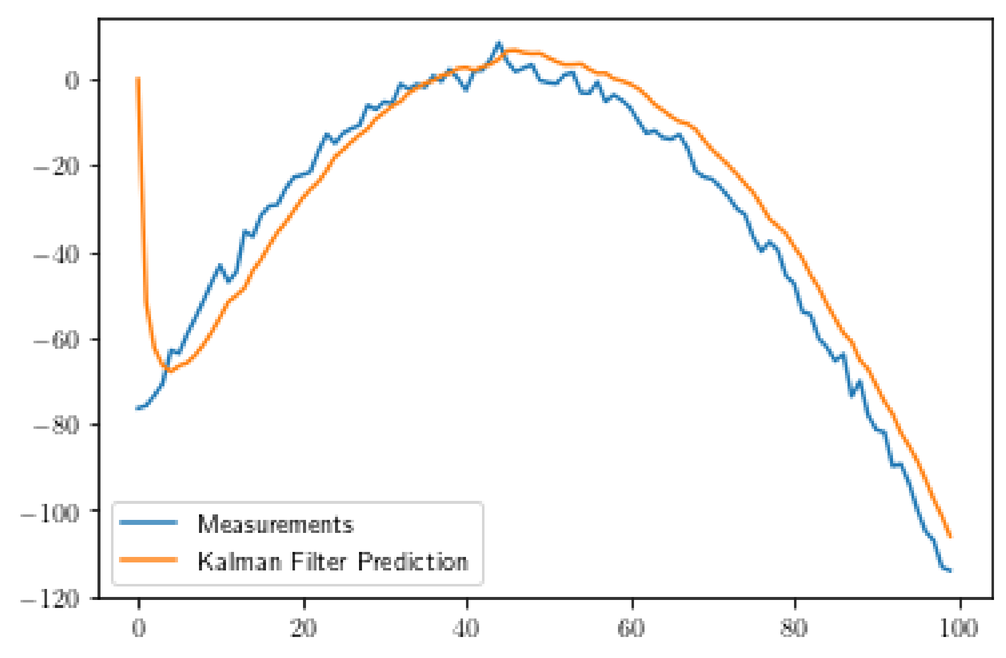
Notación

$\mathbf{x} \in \mathbb{R}^p$: State vector (may not seen)
$\mathbf{z} \in \mathbb{R}^m$: Measurement vector (what we see)
$\mathbf{u} \in \mathbb{R}^p$: Control vector (external signal)
$\mathbf{B} \in \mathbb{R}^{p\times p}$: Control matrix (can be zero)
$\mathbf{A} \in \mathbb{R}^{p\times p}$: Dynamics system matrix (state-transition model from time n to $n+1$)
$\mathbf{H} \in \mathbb{R}^{m\times p}$: Measurement matrix, how measures are acquired
$\mathbf{w} \in \mathbb{R}^p$: Process noise
$\mathbf{Q} \in \mathbb{R}^{p\times p}$: Process covariance matrix
$\mathbf{v} \in \mathbb{R}^m$: Measurement noise
$\mathbf{R} \in \mathbb{R}^{m\times m}$: Measurement covariance matrix
$\mathbf{\Sigma} \in \mathbb{R}^{p\times p}$: Covariance of the model error
$\mathbf{K} \in \mathbb{R}^{p\times m}$: Kalman Gain

El Filtro de Kalman

- Solución recursiva (IIR) al problema de filtrado lineal
- Es óptimo en el sentido de mínimos cuadrados, al minimizar la covarianza del error estimada
- Emplea dos etapas
  - **Predicción**: Predice el valor futuro usando ecuaciones de estado
  - **Corrección**: Corrige la predicción por medio de un estimador basado en mediciones

- Estima el proceso a partir de realizaciones con ruido



El Modelo Matemático

- Se asume el modelo de estado

$$\mathbf{x}(\mathrm{n}+1) = \mathbf{A} \times \mathbf{x}(n) + \mathbf{B} \times \mathbf{u}(n) + \mathbf{w}(\mathrm{n})$$

  - $\mathbf{w}(\mathrm{n})$ es una señal de ruido con covarianza $\mathbf{Q}(\mathrm{n})$

$$\mathbf{Q}(\mathrm{n}) = \mathbb{E}[\mathbf{w}^2(n)]$$

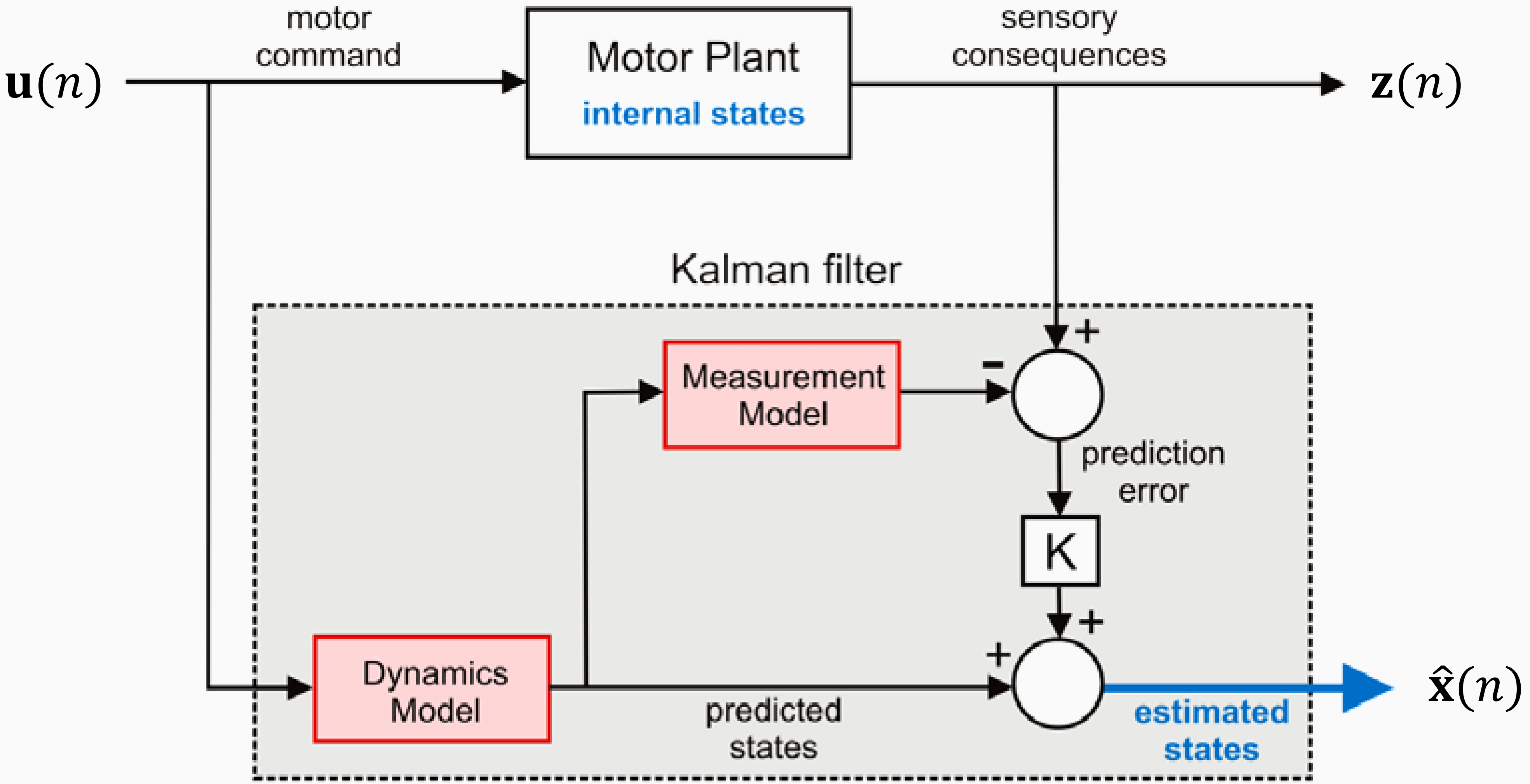
- El modelo de medición

$$\mathbf{z}(\mathrm{n}) = \mathbf{H} \times \mathbf{x}(n) + \mathbf{v}(\mathrm{n})$$

  - $\mathbf{v}(\mathrm{n})$ es una señal de ruido con covarianza $\mathbf{R}(\mathrm{n})$

$$\mathbf{R}(\mathrm{n}) = \mathbb{E}[\mathbf{v}^2(n)]$$

Diagrama a Bloques General



Initial estimates

$\hat{\mathbf{x}}(-1)$

$\hat{\boldsymbol{\Sigma}}(-1)$

$$\hat{\mathbf{x}}^{-}(n) = \mathbf{A}\hat{\mathbf{x}}(n-1) + \mathbf{B}\boldsymbol{u}(n)$$

$$\hat{\boldsymbol{\Sigma}}^{-}(n) = \mathbf{A}\hat{\boldsymbol{\Sigma}}(n-1)\mathbf{A}^{T} + \mathbf{Q}$$

Prediction

Correction

$$\mathrm{K}(n) = \hat{\boldsymbol{\Sigma}}^{-}(n)\mathbf{H}^{T}\left(\mathbf{H}\hat{\boldsymbol{\Sigma}}^{-}(n)\mathbf{H}^{T} + \mathbf{R}\right)^{-1}$$

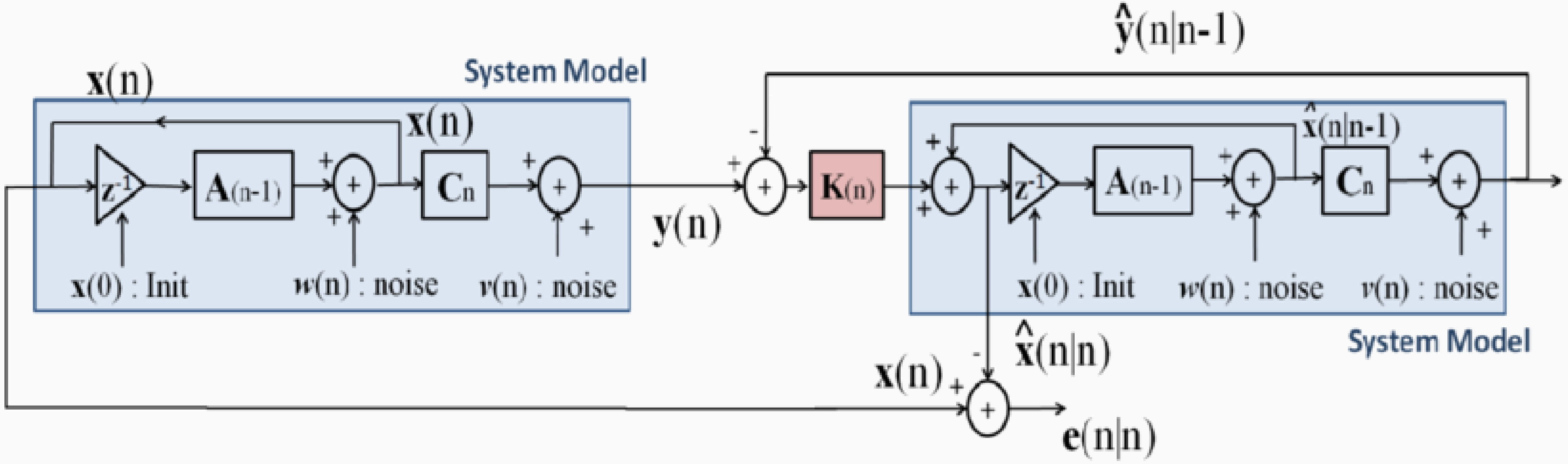
$$\hat{\mathbf{x}}(n) = \hat{\mathbf{x}}^{-}(n) + \mathrm{K}(n)\left(\mathbf{z}(n) - \mathbf{H}\hat{\mathbf{x}}^{-}(n-1)\right)$$

$$\hat{\boldsymbol{\Sigma}}(n) = \hat{\boldsymbol{\Sigma}}^{-}(n) - \mathrm{K}(n)\mathbf{H}\hat{\boldsymbol{\Sigma}}^{-}(n)$$

$\hat{\boldsymbol{\Sigma}}^{-}$: Prediction

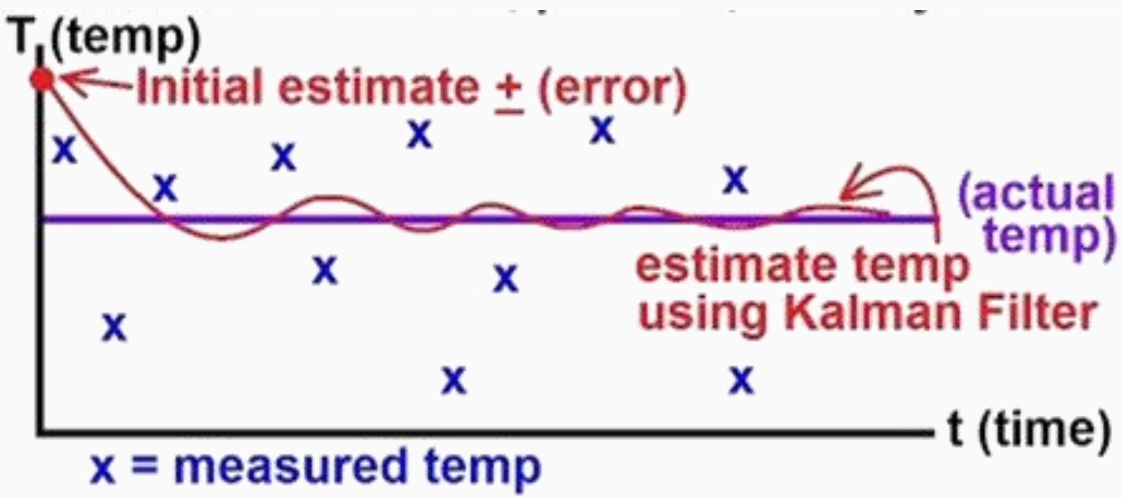
$\hat{\boldsymbol{\Sigma}}$: Estimation

Diagrama a Bloques



Convergencia

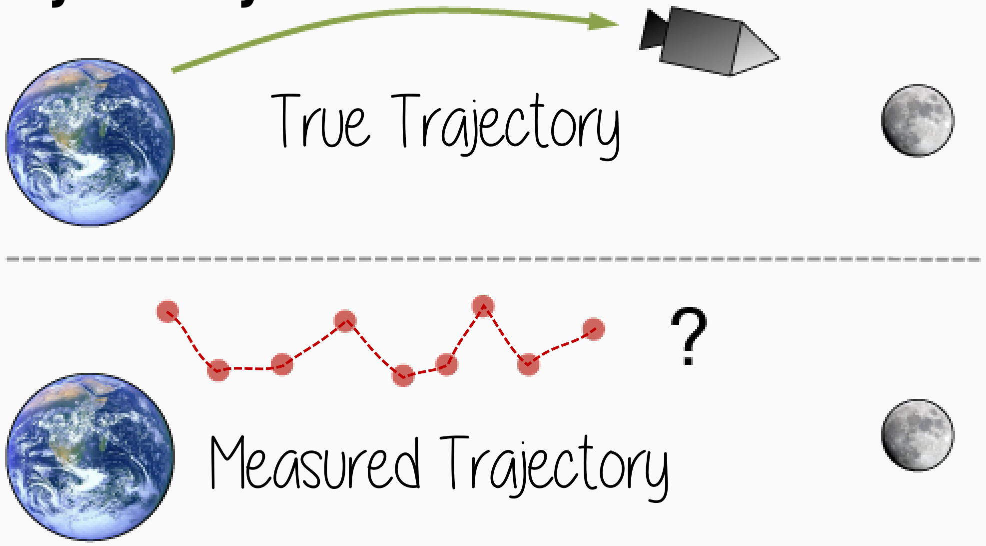
- Si el sistema esta en reposo, el filtro de Kalman converge al valor verdadero con el tiempo



For instance, on a cartesian 2D plane, if we are only capable of measuring the trajectory of a starship

$$\mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix} \in \mathbb{R}^2$$

However, sensors are not perfect, and they always contains errors



The State Vector

$$\mathbf{x}(n) = \begin{bmatrix} x \\ y \\ v_x \\ v_y \\ a_x \\ a_y \end{bmatrix} \in \mathbb{R}^6$$

Then, $p = 6$ and $m = 2$

$$\mathbf{A} \in \mathbb{R}^{6\times 6}$$

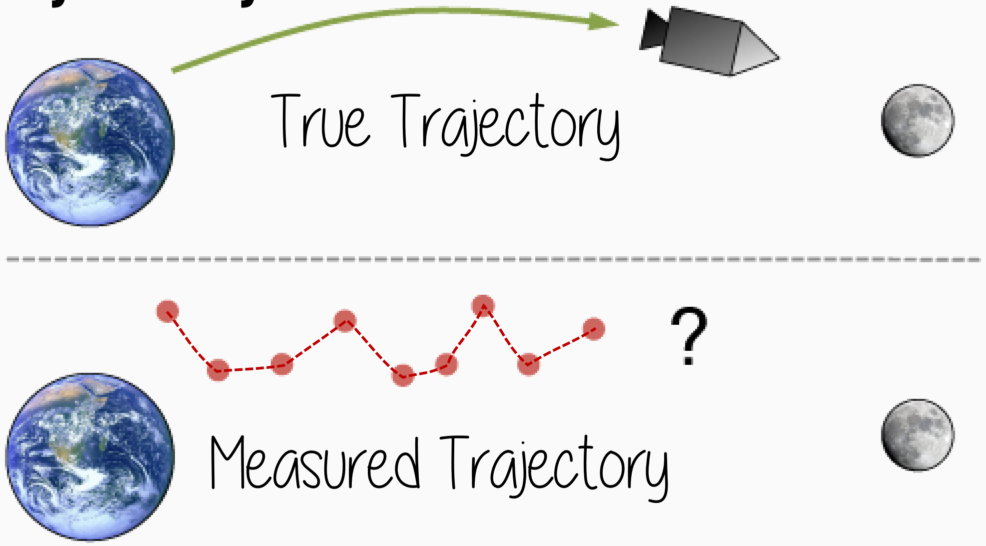
$$\mathbf{H} \in \mathbb{R}^{2\times 6}$$

We also assume no external control

For instance, on a cartesian 2D plane, if we are only capable of measuring the trajectory of a starship

$$\mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

However, sensors are not perfect, and they always contains errors



System Dynamics Model: Assuming constant acceleration

- Horizontal Dynamics

$$\ddot{\mathbf{x}}\,(n+1) = \ddot{\mathbf{x}}(n)$$

$$\dot{\mathbf{x}}(n+1) = \dot{\mathbf{x}}_x(n) + \Delta t \ddot{\mathbf{x}}_x(n)$$

$$x(n+1) = x(n) + \Delta t \dot{\mathbf{x}}_x(n) + \frac{(\Delta t)^2}{2} \ddot{\mathbf{x}}_x(n)$$

- Vertical Dynamics

$$\ddot{\mathbf{x}}_y(n+1) = \ddot{\mathbf{x}}_y(n)$$

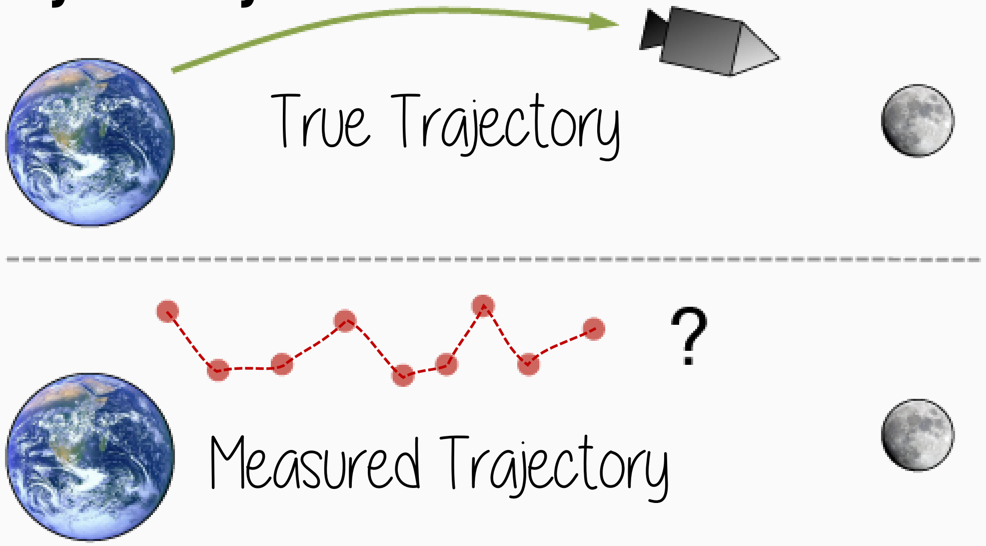
$$\dot{\mathbf{x}}_y(n+1) = \dot{\mathbf{x}}_y(n) + \Delta t \ddot{\mathbf{x}}_y(n)$$

$$y(n+1) = y(n) + \Delta t \dot{\mathbf{x}}_y(n) + \frac{(\Delta t)^2}{2} \ddot{\mathbf{x}}_y(n)$$

For instance, on a cartesian 2D plane, if we are only capable of measuring the trajectory of a starship

$$\mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

However, sensors are not perfect, and they always contains errors



Dynamical System:

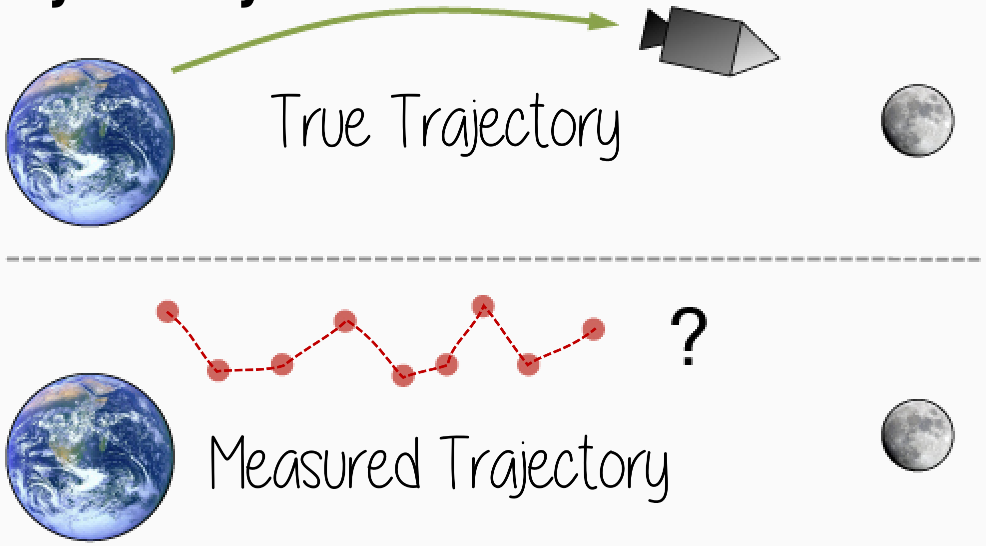
- Project onto the next state.
- Represents the system's equations into a matrix
- Each column is a variable of $\mathbf{x}(\mathrm{n})$
- Each row is a system equation

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & .5\Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & .5\Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

For instance, on a cartesian 2D plane, if we are only capable of measuring the trajectory of a starship

$$\mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

However, sensors are not perfect, and they always contains errors



The first row is associated to $x(n+1)$

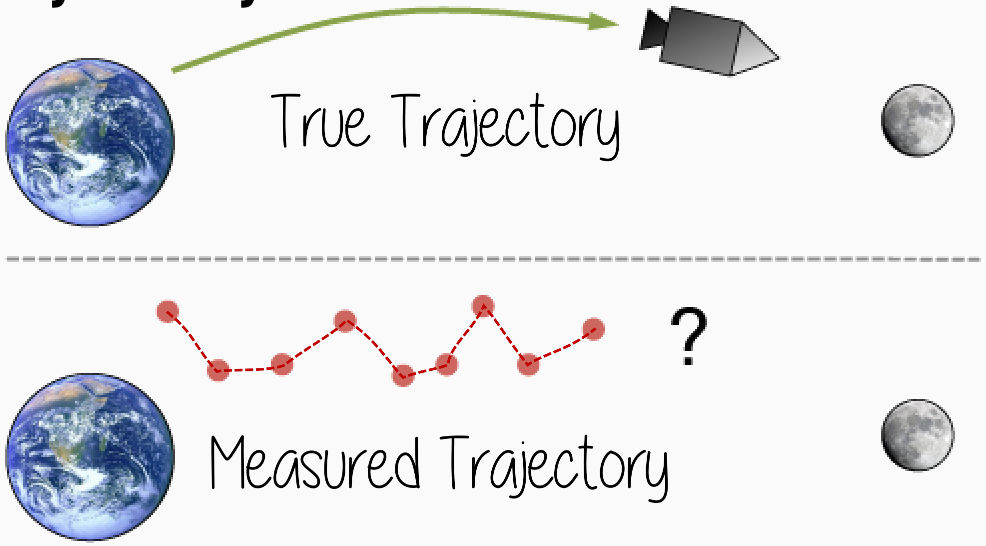
$$\begin{bmatrix} 1 & 0 & \Delta t & 0 & .5\Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & .5\Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ \dot{\mathbf{x}}_x \\ \dot{\mathbf{x}}_y \\ \ddot{\mathbf{x}}_x \\ \ddot{\mathbf{x}}_y \end{bmatrix}$$

$$x(n+1)$$
$$= 1 \cdot x(n) + \Delta t \dot{\mathbf{x}}_x(n) + \frac{\Delta t^2}{2} \ddot{\mathbf{x}}_x(n)$$

For instance, on a cartesian 2D plane, if we are only capable of measuring the trajectory of a starship

$$\mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

However, sensors are not perfect, and they always contains errors



The measurement matrix translates from the state vector to the measurement vec.

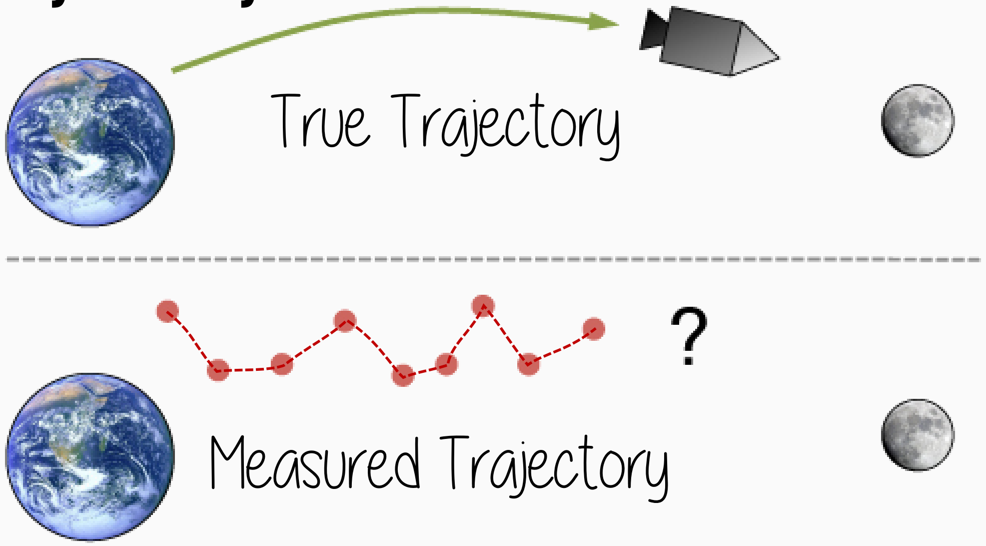
$$\overbrace{\begin{bmatrix} ? & ? & ? & ? & ? & ? \\ ? & ? & ? & ? & ? & ? \end{bmatrix}}^{\mathbf{H}} \begin{bmatrix} x \\ y \\ \dot{\mathbf{x}}_x \\ \dot{\mathbf{x}}_y \\ \ddot{\mathbf{x}}_x \\ \ddot{\mathbf{x}}_y \end{bmatrix}$$

$$= \mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

For instance, on a cartesian 2D plane, if we are only capable of measuring the trajectory of a starship

$$\mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

However, sensors are not perfect, and they always contains errors



The measurement matrix translates from the state vector to the measurement vec.

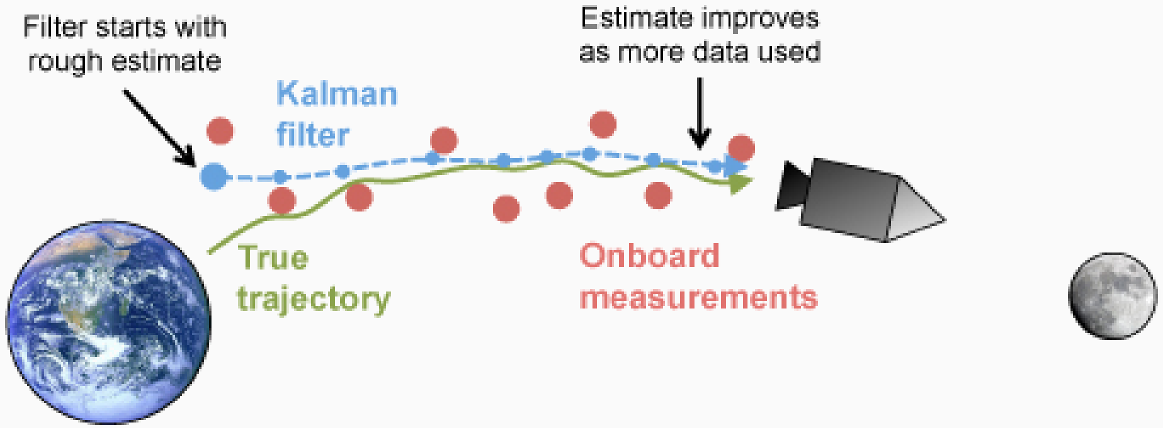
$$\overbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}}^{\mathbf{H}} \begin{bmatrix} x \\ y \\ \dot{\mathbf{x}}_x \\ \dot{\mathbf{x}}_y \\ \ddot{\mathbf{x}}_x \\ \ddot{\mathbf{x}}_y \end{bmatrix}$$

$$= \mathbf{z}(n) = \begin{bmatrix} x \\ y \end{bmatrix}$$

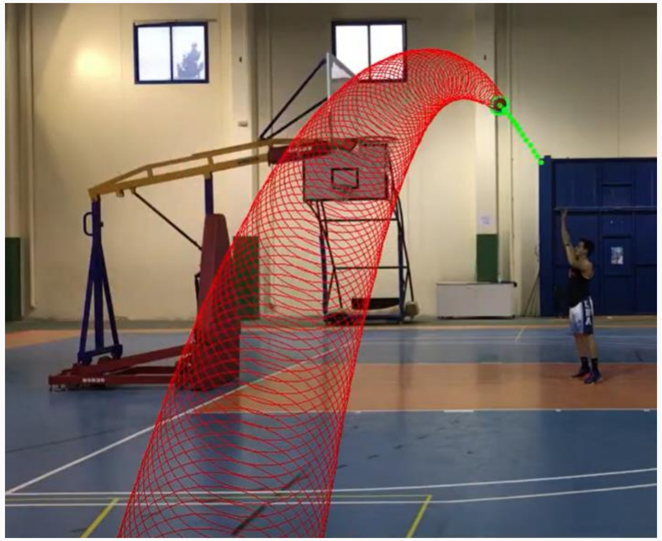
Aplicaciones

- Rastreo de objetos
  - Misiles
  - Caras
  - Manos
- Economía
  - Predicción de bolsa de valores
- Navegación
  - GPS
  - IMUs
- Visión por Computadora
  - Mediciones de profundidad
  - Rastreo de características
  - Fusión de datos (e.g., GPS + IMU)
  - $\vdots$

Aplicaciones



Object tracking



Object tracking

Trajectory planning on quadcopters

$\mathbf{B} \neq \mathbf{0}$

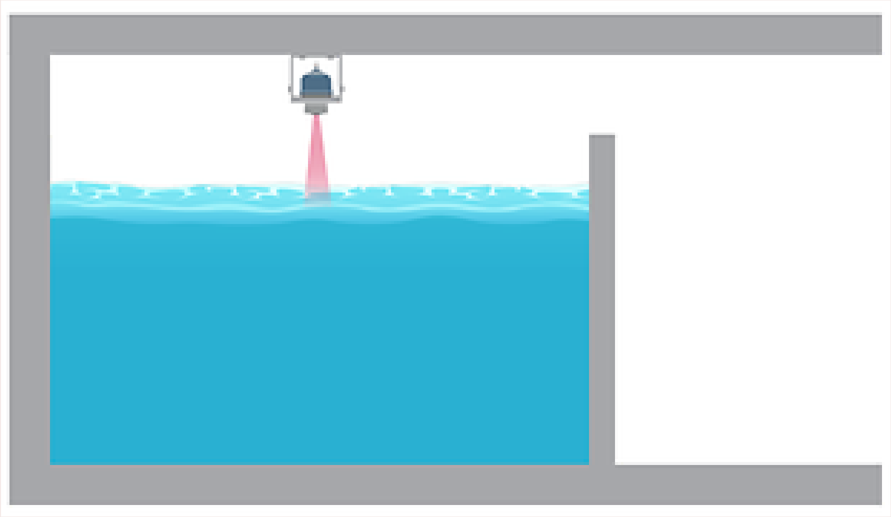
$\boldsymbol{u}$: Acceleration (control signal)

$\mathbf{z}$: 3D Position

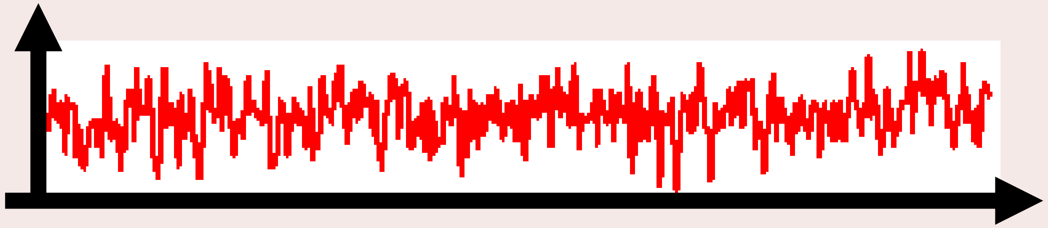
$\mathbf{x}$: Position and elevation

Ejemplo: Predicción de Nivel de Agua de un Tanque

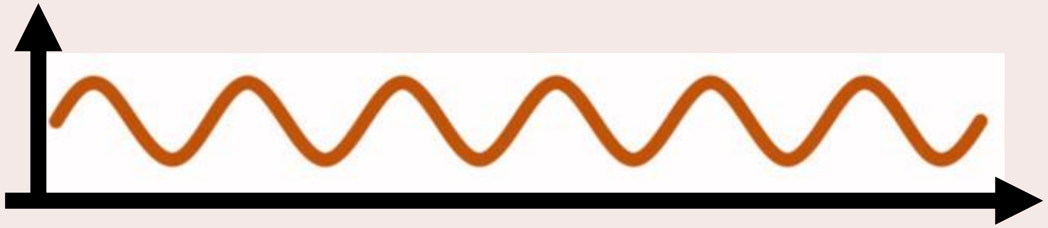
- Mediciones ruidosas debido a la turbulencia del agua



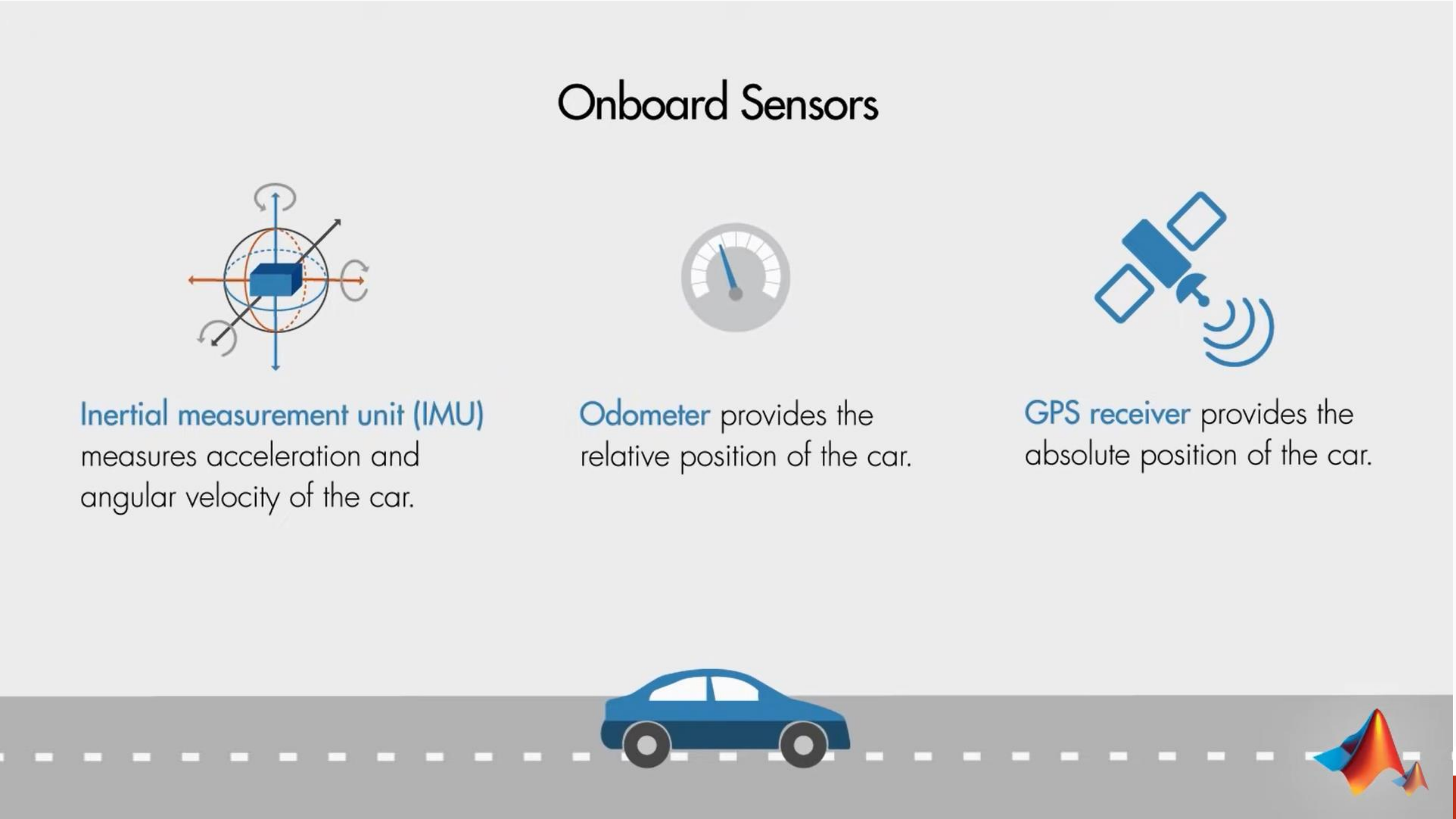
- El sensor introduce ruido



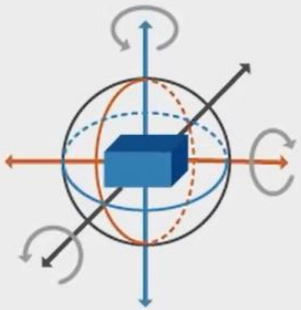
- El proceso introduce ruido por las perturbaciones físicas



Ejemplo: Localización de un vehículo



Ejemplo: Localización de un vehículo



IMU

Odometer

GPS

Measure the relative position of the car

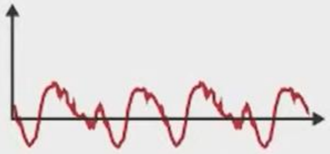
Update frequency FAST

Prone to drift

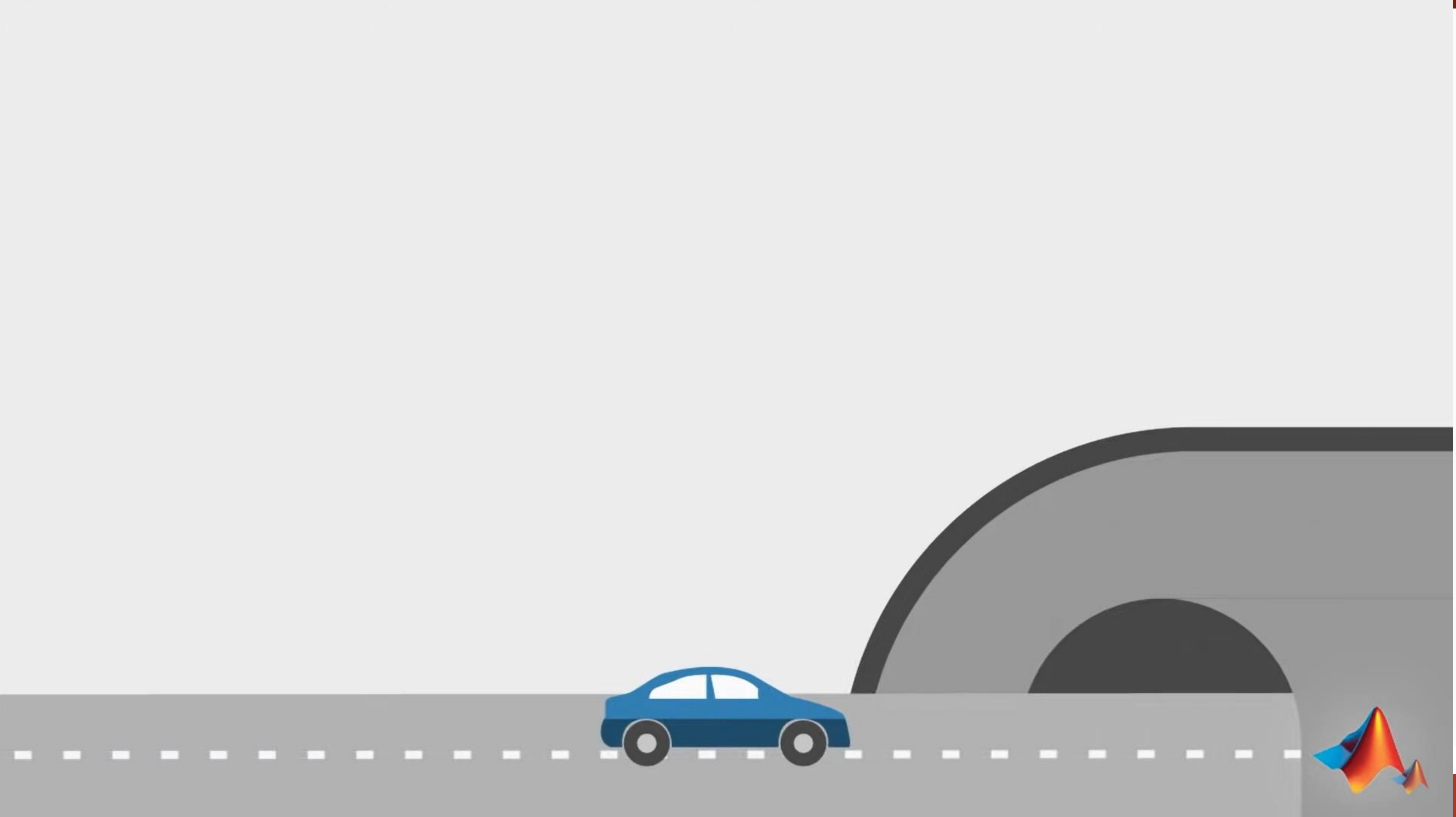
Measures the absolute position of the car

Update frequency SLOW

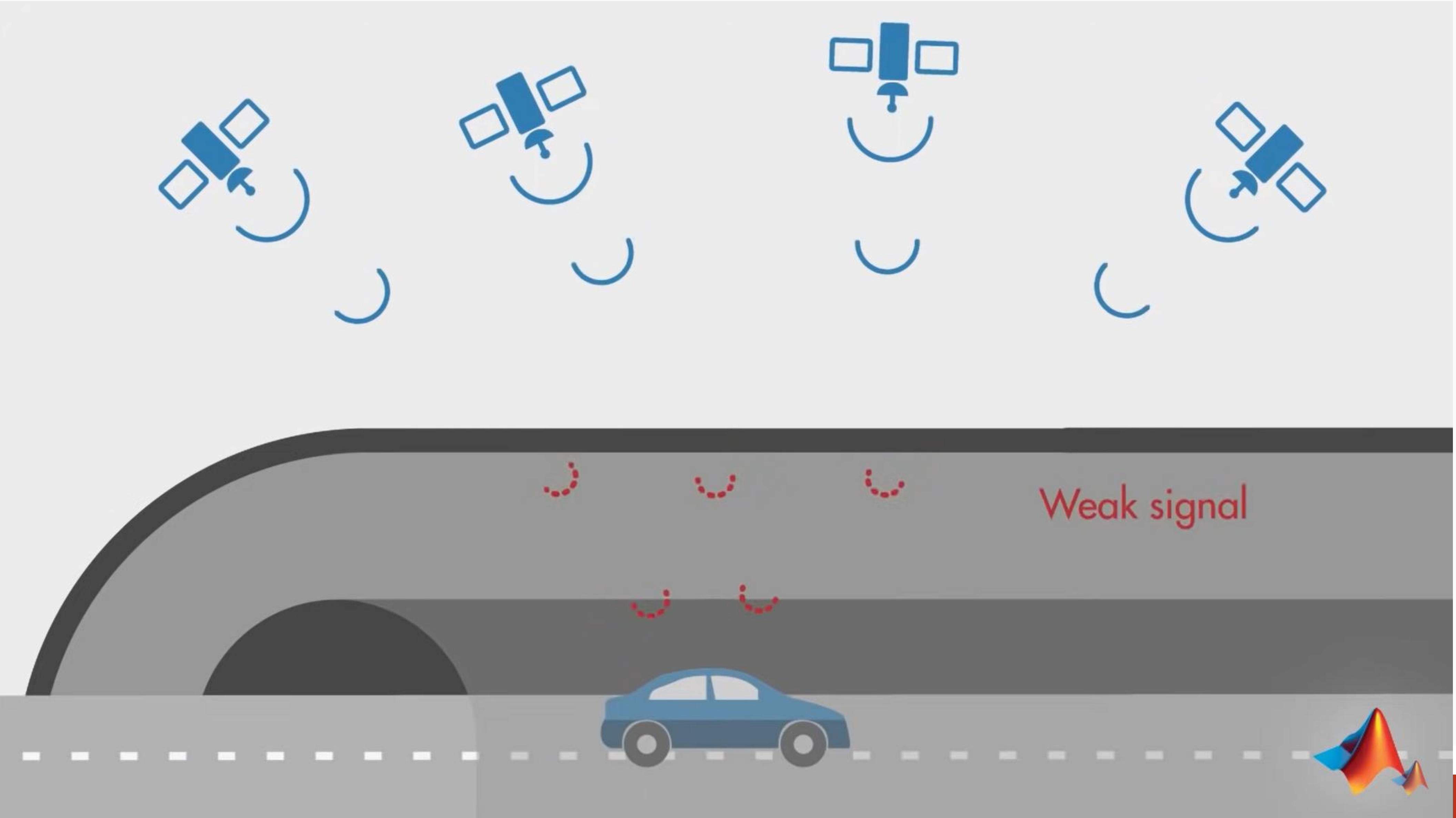
Noisy



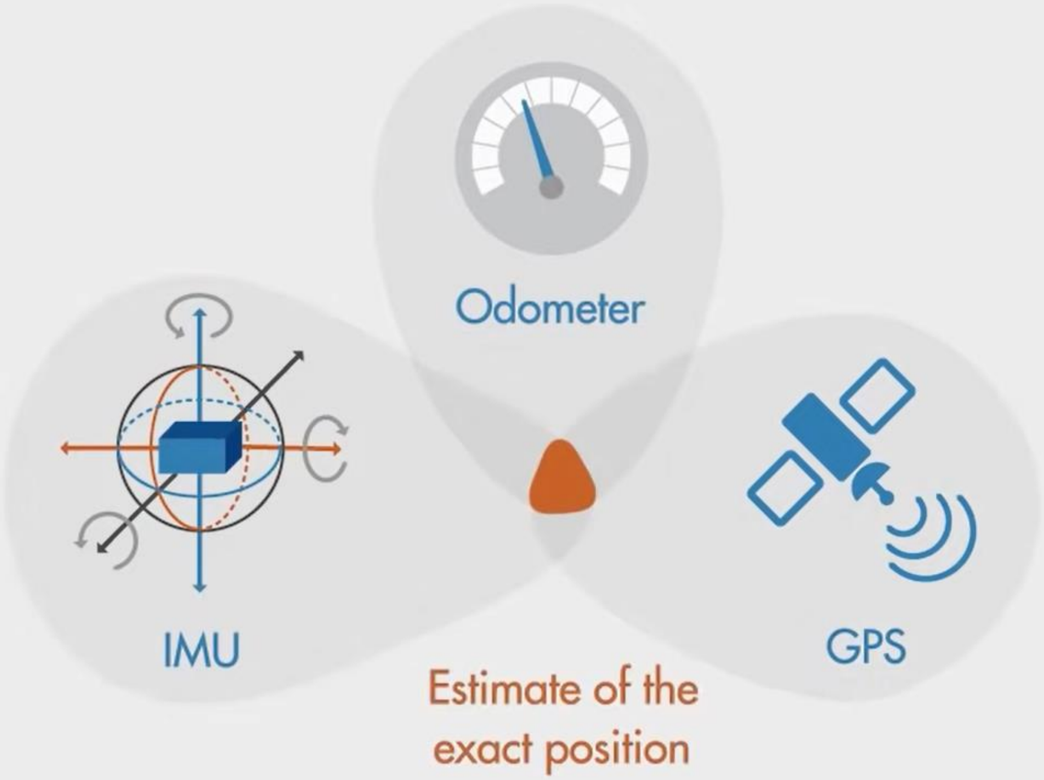
Ejemplo: Localización de un vehículo



Ejemplo: Localización de un vehículo

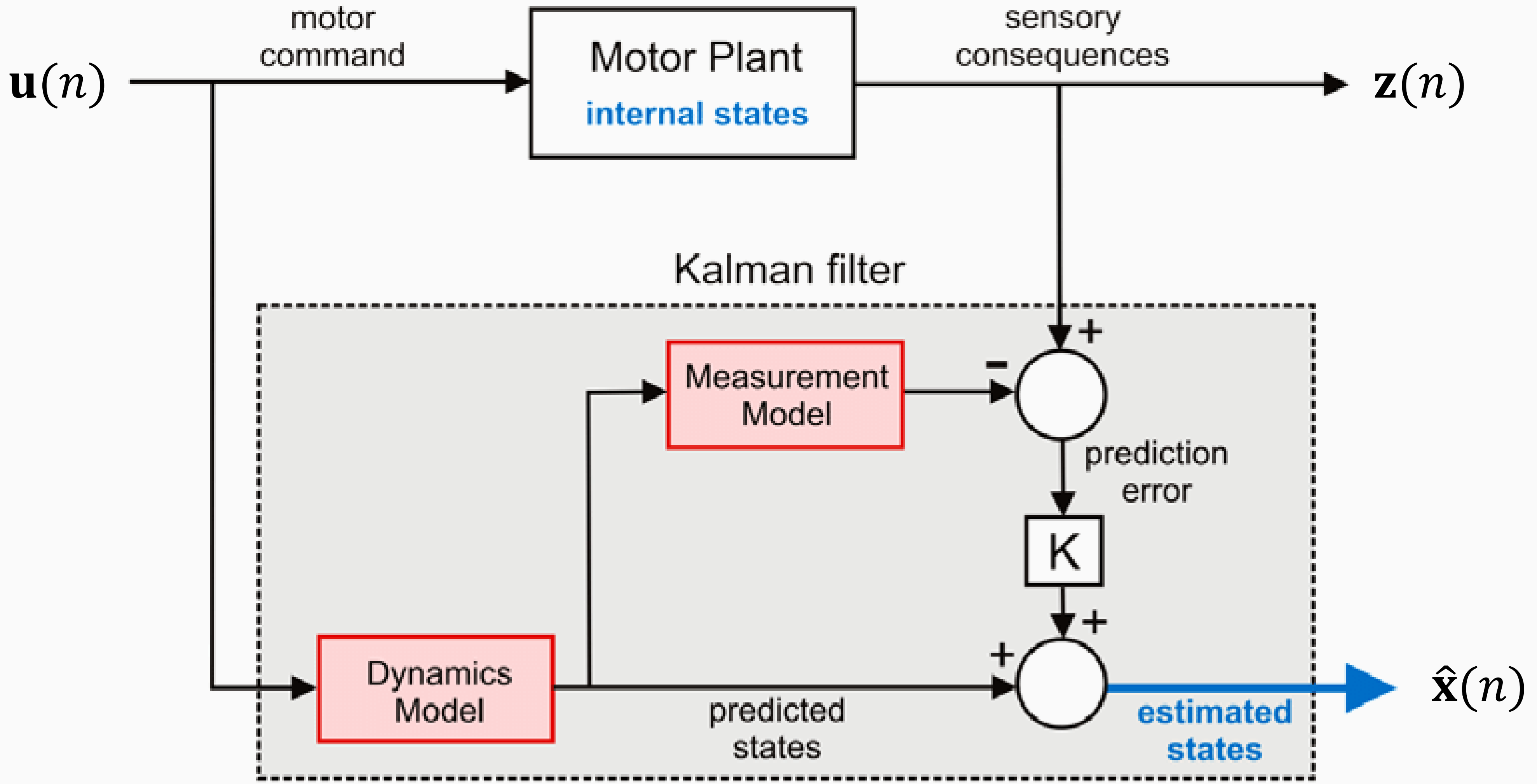


Ejemplo: Localización de un vehículo



Laboratorio 10. Filtro de Kalman

Esquema General



Actividad: Implementar Un Sistema de Seguimiento de Objetos en Video

1. En el repositorio, descargue el archivo Kalman.py, BallDetector.py y script.py
2. Implemente las ecuaciones del filtro de Kalman en el archivo Kalman.py para los métodos de predict y correct
3. En el archivo de script, aplique el filtro de Kalman para realizar el seguimiento de la pelota en el video ball.mp4

- Crear matrices, por ejemplo para una matriz de 2x2
  - np.array([[1,2], [3,4]])
- Multiplicaciones matriciales se realizan con np.dot(A,B)
  - Por ejemplo si se desea calcular $C = A \times B$, en Python sería
  - C = np.dot(A,B)
- Transpuesta (dos opciones)
  - np.transpose(A)
  - A.T
- Inversa: np.linalg.inv(A)

Actividad: Implementar Un Sistema de Seguimiento de Objetos en Video

1. En el repositorio, descargue el archivo Kalman.py, BallDetector.py y script.py
2. Implemente las ecuaciones del filtro de Kalman en el archivo Kalman.py para los métodos de predict y correct
3. En el archivo de script, aplique el filtro de Kalman para realizar el seguimiento de la pelota en el video ball.mp4

- Para el paso 3 asuma el modelo de velocidad constante
- $x(n+1) = x(n) + \Delta t * v_x(n)$
- $v_x(n+1) = v_x(n)$