TECNICHE DI CONTROLLO MULTIVARIABILE

Esempi di applicazioni ingegneristiche

Filtro di Kalman

- Il **filtro di Kalman** è trova molteplici applicazioni in svariati settori, per citarne alcuni:
  - Sistemi di controllo,
  - Computer vision,
  - Localizzazione e navigazione,
  - Econometria…

- Si tratta di uno strumento molto flessibile, utilizzato anche qualora perda le sue caratteristiche teoriche di ottimalità, in particolare:
  - Può essere utilizzato come algoritmo di **sensor-fusion,**
  - Può essere utilizzato con dispositivi che funzionano a differenti tempi di campionamento (**multi-rate**),
  - Può fornire una stima anche in momenti in cui le misure non sono disponibili.

Object tracking con filtro di Kalman

- Descrizione applicazione:

Si vuole stimare posizione, velocità e accelerazione di un oggetto movimentato da un nastro trasportatore.

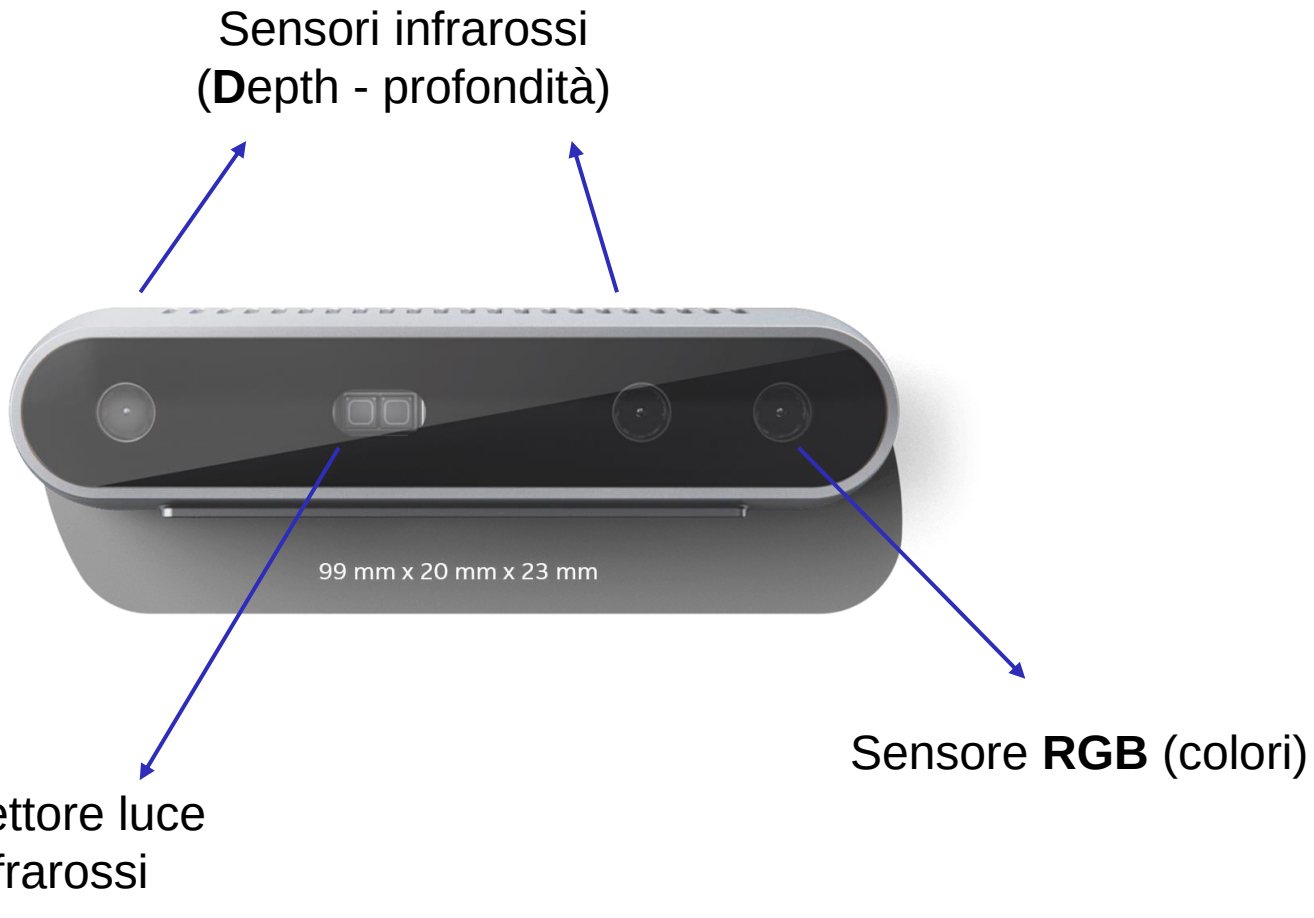
- Strumentazione (da laboratorio LiraLab):
  - Nastro trasportatore
  - Telecamera RGB-D
  - Marker Aruco

Nastro Trasportatore

- Piccolo nastro trasportatore da laboratorio (1.20 m di lunghezza) movimentato da motore DC 24V

Telecamera RGBD

- Intel Realsense D415

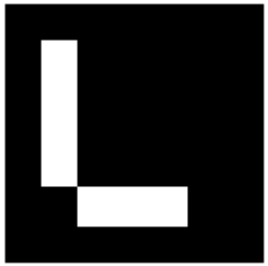


Aruco Markers e OpenCV

- Marker per comupter vision:

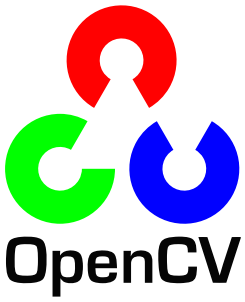
ID = 1

ID = 2

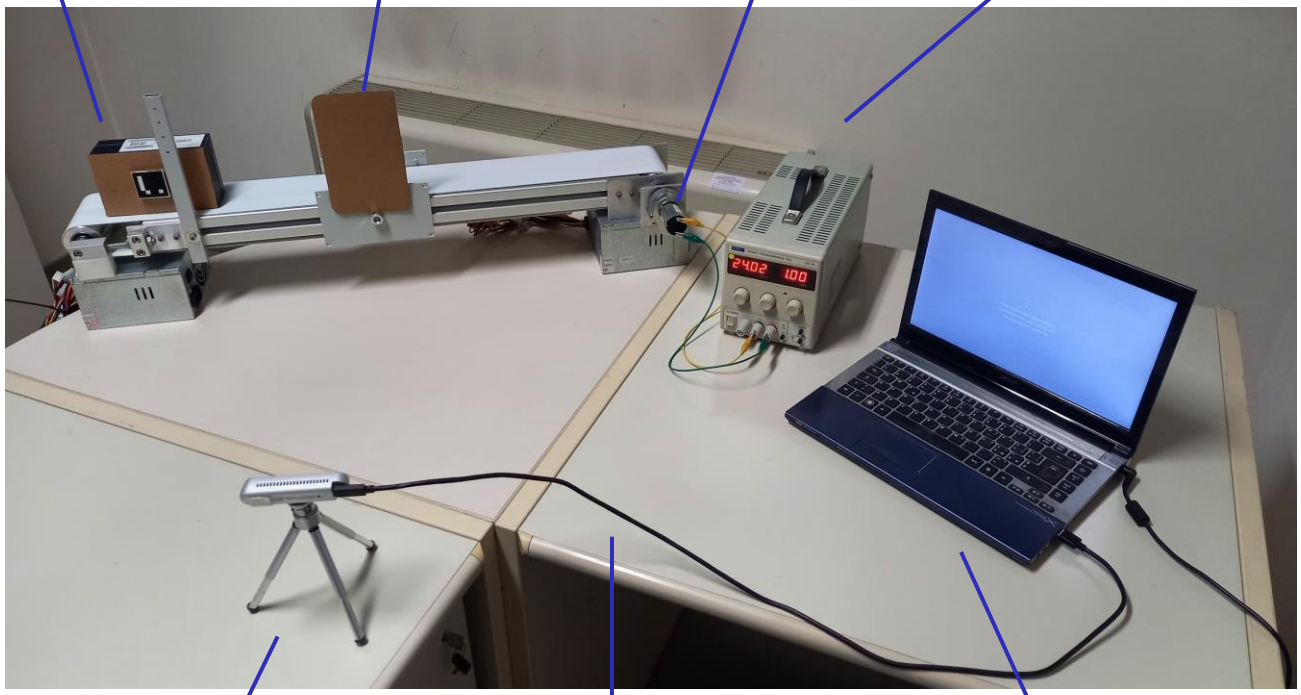


ID = 3

- OpenCV: libreria open source (c++) per computer vision
  - Implementa molteplici algoritmi di visione artificiale fra cui il riconoscimento dei marker Aruco



Setup sperimentale



Modello del sistema

- Sistema libero di ordine 9 dove lo stato è dato da posizione, velocità e accelerazione dell'oggetto lungo i tre assi cartesiani x,y,z del sistema di riferimento della telecamera
- Il modello tempo discreto utilizzato suppone che tra un istante e il successivo l'accelerazione rimanga pressoché costante

$$\boldsymbol{x} = \begin{bmatrix} p_x \\ v_x \\ a_x \\ p_y \\ v_y \\ \dots \\ a_z \end{bmatrix} \qquad \boldsymbol{x}_k = \begin{bmatrix} 1 & T & \frac{1}{2}T^2 & 0 & \dots & 0 \\ 0 & 1 & T & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix} \begin{bmatrix} p_x \\ v_x \\ a_x \\ p_y \\ v_y \\ \dots \\ a_z \end{bmatrix}_{k-1} + \boldsymbol{w}$$

Modello del sistema

- Il vettore delle misure coincide con la posizione spaziale dell'oggetto (rappresentata dal pixel identificato come vertice superiore destro del marker Aruco)

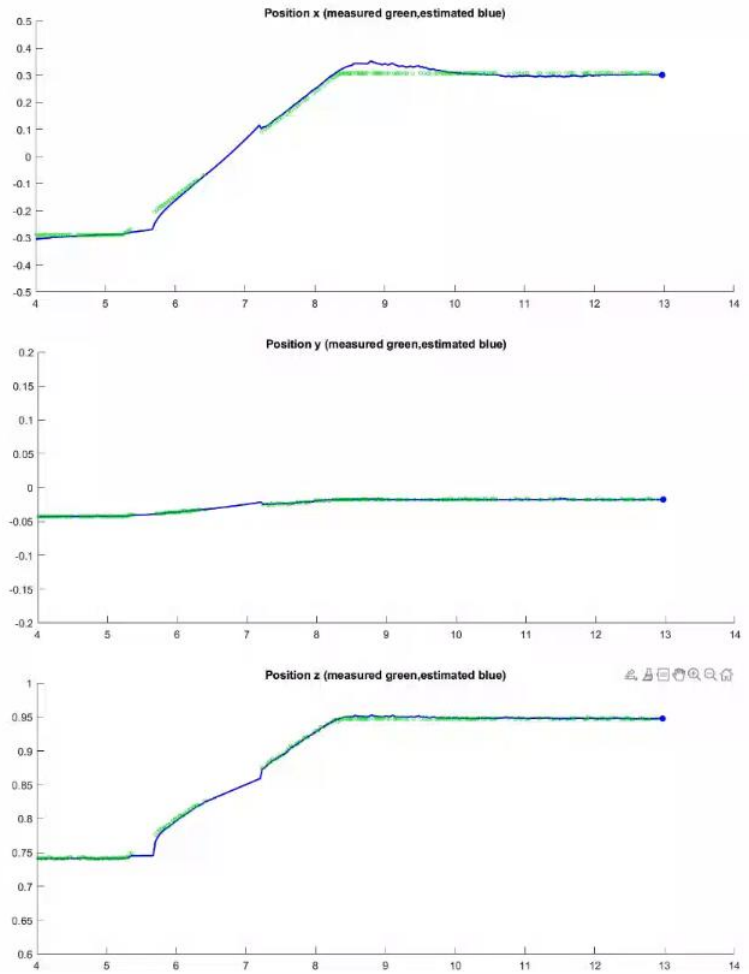
$$\boldsymbol{y} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \end{bmatrix} \boldsymbol{x} + \boldsymbol{v}$$

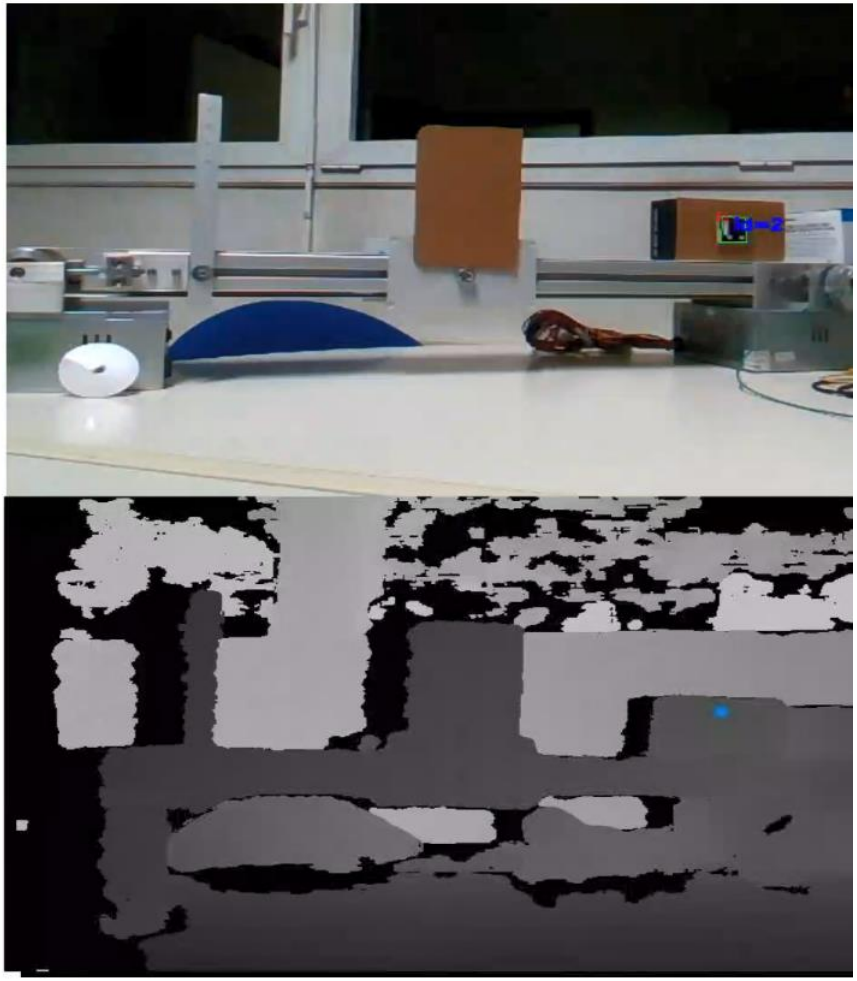
Modello del sistema

- Rumore di processo e rumore di misura sono supposti e trattati come Gaussiani ma in realtà non lo sono
  - Il rumore di processo comprende tutto ciò che rende il moto dell'oggetto non uniformemente accelerato (tra un istante e il successivo)
  - Il rumore di misura comprende gli errori di allineamento tra i frame acquisiti dai tre diversi sensori ottici e le incertezze nel processo di elaborazione della point cloud e marker detection

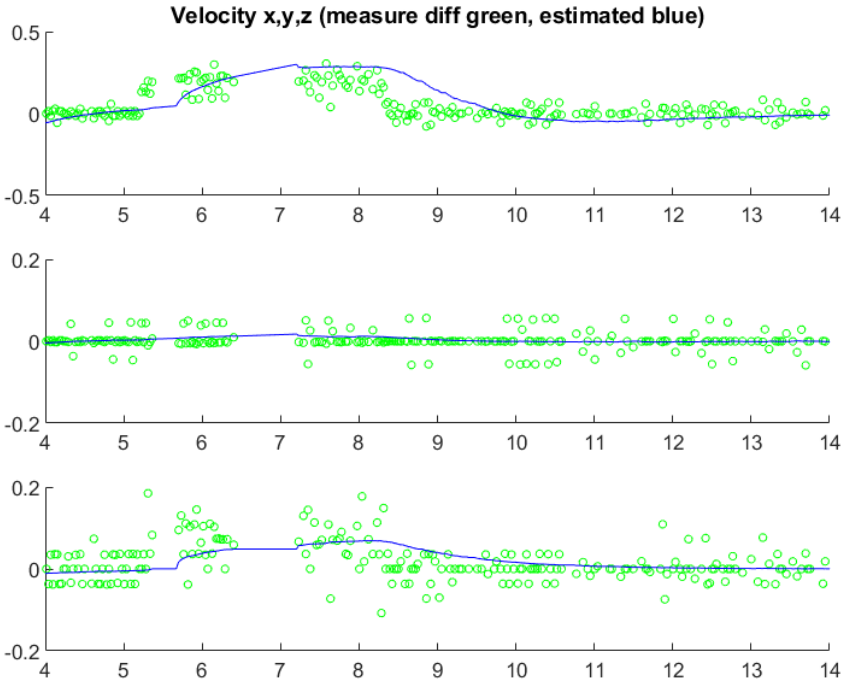
- Nei momenti in cui le **misure non sono disponibili** (perché il marker è oscurato o non è possibile calcolare la profondità del pixel individuato nella point-cloud) si può pensare di utilizzare unicamente il modello del sistema per fornire una stima dello stato, **senza la relativa correzione delle misure**

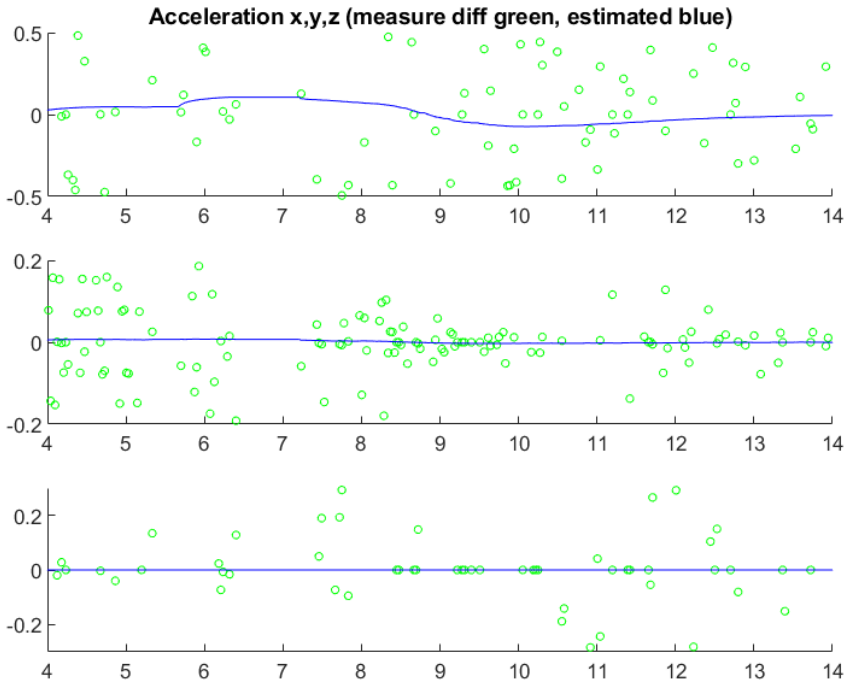
Risultati – Tracking posizione





Risultati – Tracking velocità accelerazione





Feedback Linearization in robotica

- Modello dinamico del manipolatore

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + D\dot{q} + g(q) = \tau$$

- Legge di controllo linearizzante (cancellazione delle non linearità tramite feesdback):

$$\tau = M(q)v + C(q,\dot{q})\dot{q} + D\dot{q} + g(q) \Rightarrow \ \ddot{q} = v$$

- Inseguimento di una traiettoria desiderata:

$$v = \ \ddot{q}_d - K_d(\dot{q} - \dot{q}_d) - K_p(q - q_d)$$

Robot manipolatore

- Franka Emika Panda Robot:
  - 7 assi (giunti motorizzati)
  - Sensori di coppia ai giunti
  - Payload massimo 3 Kg
  - Utilizzato per applicazioni in cui è richiesta destrezza e collaborazione con operatori (Human – Robot Interaction)
  - Programmazione tramite Franka – Desk o ROS (Robot Operating System)

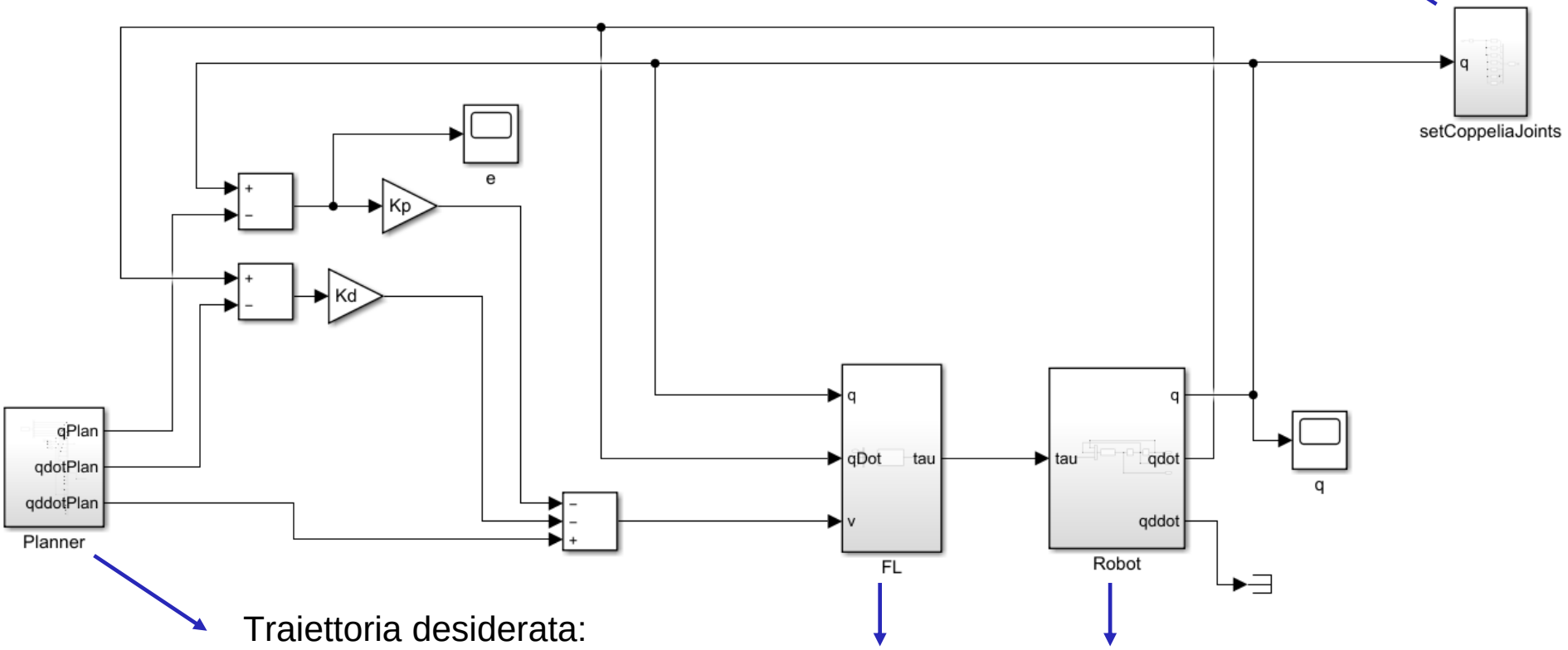
Software per la simulazione

- Robotics toolbox for **Matlab** by Peter Corke:
  - Funzionalità per l'analisi e la simulazione di robot manipolatori (cinematica, dinamica, generazione di traiettorie…)
  - Contiene modelli cinematici e dinamici di svariati robot manipolatori, tra questi il Franka Emika Panda robot

- **CoppeliaSim**:
  - Software (gratuito in versione EDU) per la simulazione di complesse scene dinamiche
  - Svariati modelli di manipolatori (tra cui il Franka Emika Panda robot)
  - Ambiente di sviluppo integrato (linguaggio LUA)
  - Possibilità di interfacciarsi con applicazioni esterne (anche Matlab)

Schema di controllo

Funzioni di interfaccia con CoppeliaSim (setting della posizione dei giunti)



Traiettoria desiderata: giunto 2 sinusoidale, gli altri giunti costanti

Robotics Toolbox: dinamica e dinamica inversa del manipolatore

Simulazione

Scena CoppeliaSim

Simulink: scope sulla posizione dei giunti

Simulink: scope sull'errore di tracking