Corso di Robotica 2

Asservimento visuale

Prof. Alessandro De Luca

DIPARTIMENTO DI INFORMATICA
E SISTEMISTICA ANTONIO RUBERTI

SAPIENZA
UNIVERSITÀ DI ROMA

Visual Servoing

- obiettivo

  usare informazioni acquisite tramite sensori di visione (telecamere) per controllare la posa/moto di un robot (o di una parte di esso)

+ acquisizione dati ~ occhio umano, altissimo contenuto informativo nelle immagini acquisite

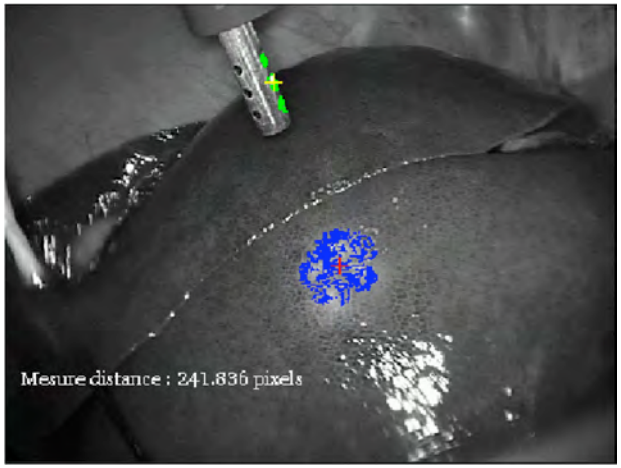
– difficoltà nell'estrarre i dati essenziali, trasformazioni prospettiche non-lineari, alta sensibilità alle condizioni ambientali (illuminazione), rumore

Applicazioni

navigazione automatica per sistemi robotici (agricoltura, automotive)

video-sorveglianza

sincronizzazione movimenti (robotica chirurgica)



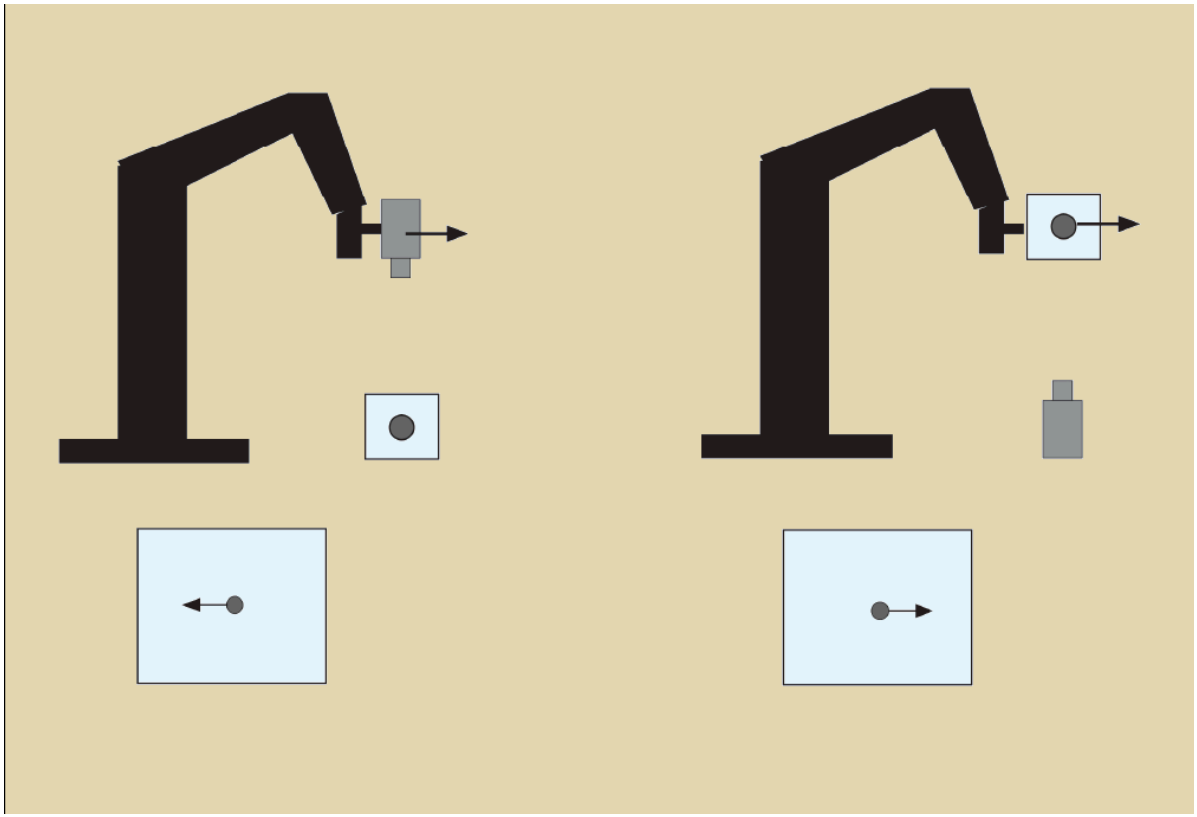
Elaborazione delle immagini

- estrazione di parametri caratteristici utili per il controllo in tempo reale
  - features: punti, baricentro, area, momenti superiori, ...
- elaborazione di basso livello
  - binarizzazione (soglia di grigio), equalizzazione (istogrammi), ...
- segmentazione
  - basata su regioni (eventualmente su immagine binaria)
  - basata sui bordi
- interpretazione
  - associazione dei parametri caratteristici
  - problema della corrispondenza di caratteristiche di oggetti in diverse immagini (stereo o su "flusso" di immagini)

Configurazione sistema visuale

- una, due o più telecamere
  - livelli di grigio o colore
- visione 3D o stereo
  - anche con una sola telecamera (mobile), con oggetto ripreso da due diverse posture (note)
- posizionamento telecamera
  - fissa (eye-to-hand)
  - mobile sul manipolatore (eye-in-hand)
- teste di visione robotizzate
  - motorizzate (es., pan-tilt su robot mobile Magellan)

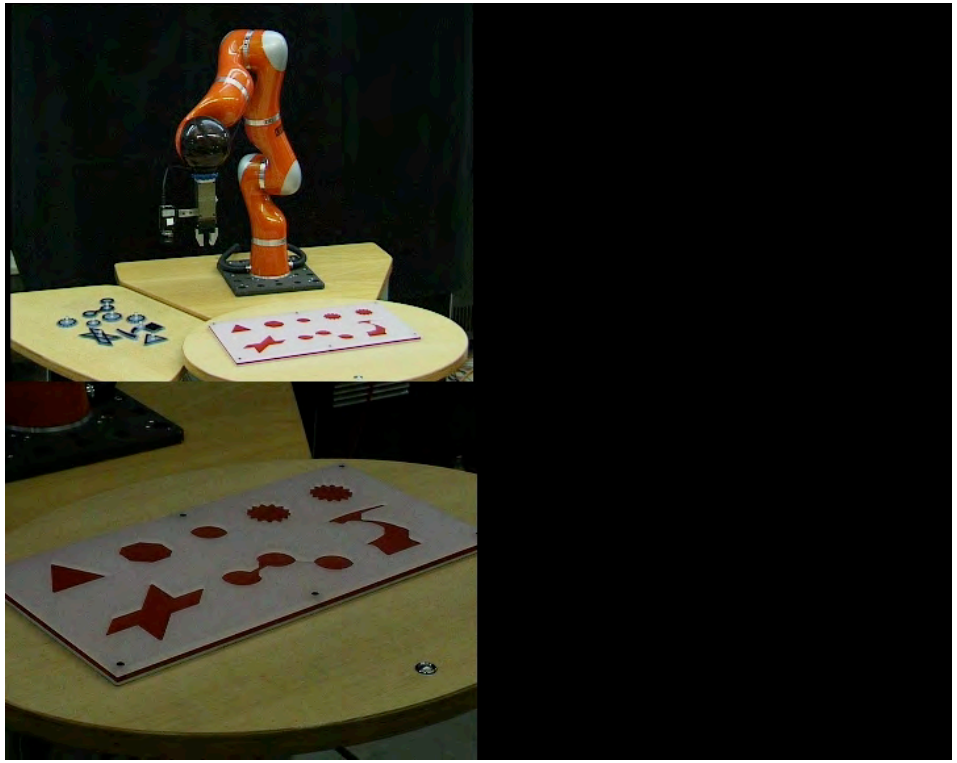
Posizionamento della telecamera

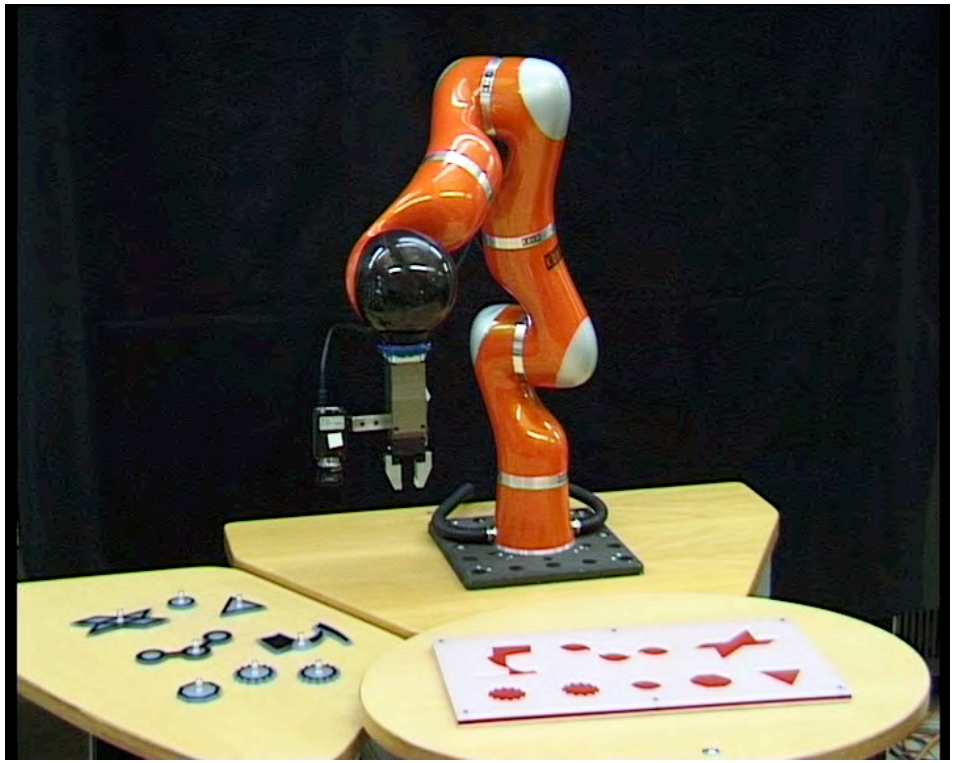


eye-in-hand

eye-to-hand

Assemblaggio con visione

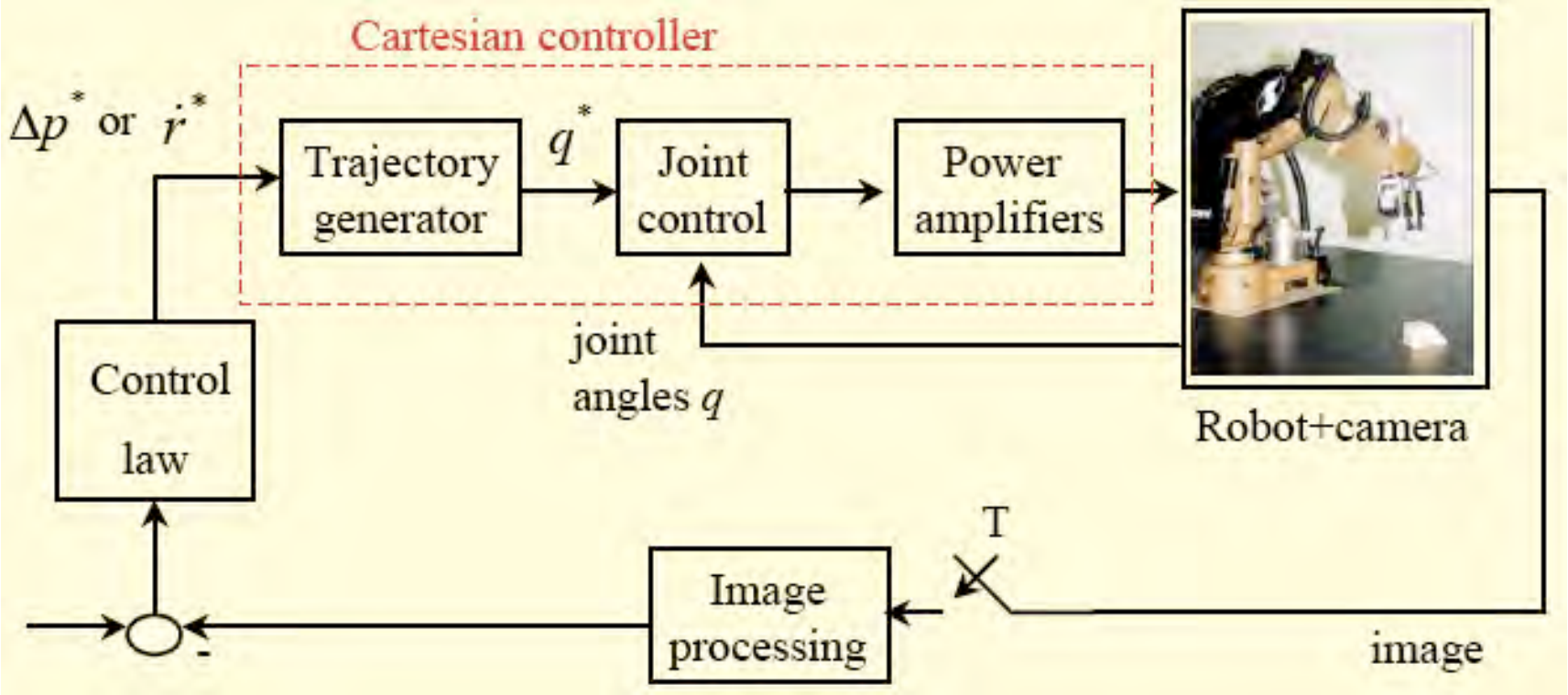




sistema PAPAS-DLR
(eye-in-hand, ibrido forza-visione)

robusto rispetto a
movimentazione del target

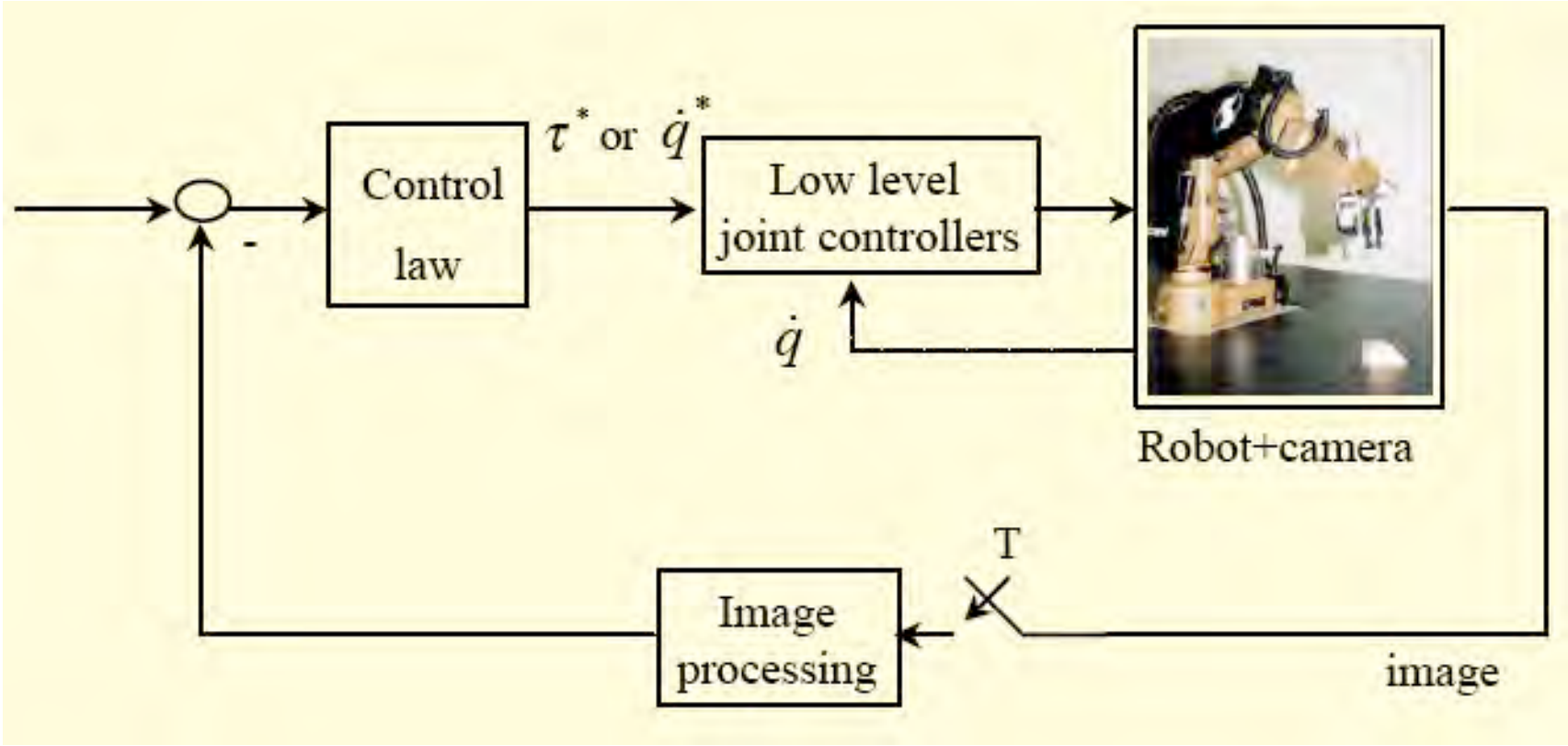
Visual servoing indiretto/esterno



sistema di visione usato per fornire i riferimenti al controllore cartesiano

- legge di controllo "facile" (la stessa del caso senza visione)
- adatto a situazioni lente (frequenza di controllo < 50Hz)

Visual servoing diretto/interno



sostituisce il controllore cartesiano con uno basato sulla visione che calcola direttamente gli ingressi ai giunti

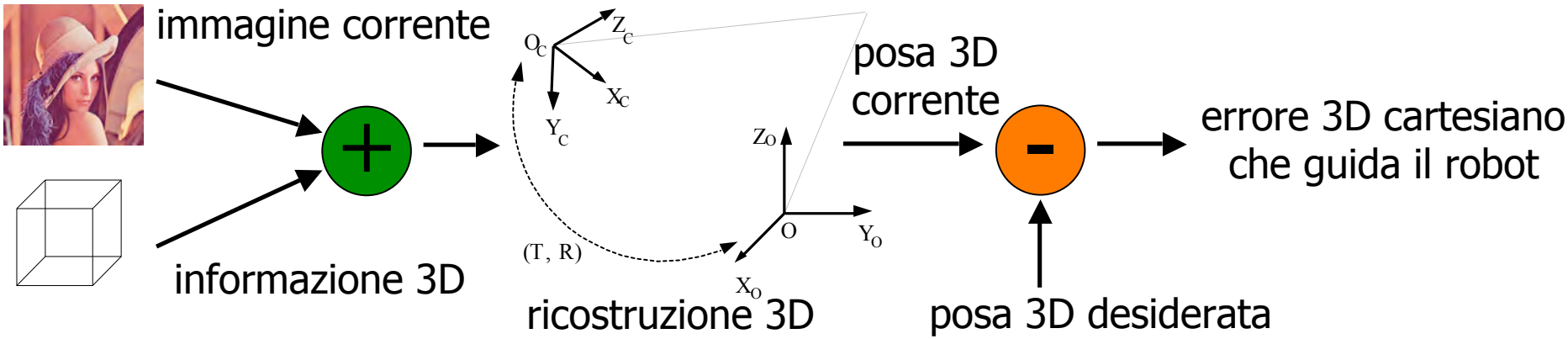
- legge di controllo più difficile (deve tener conto della dinamica del robot)
- adatto a situazioni veloci (frequenza di controllo > 50Hz)

Classificazione degli approcci

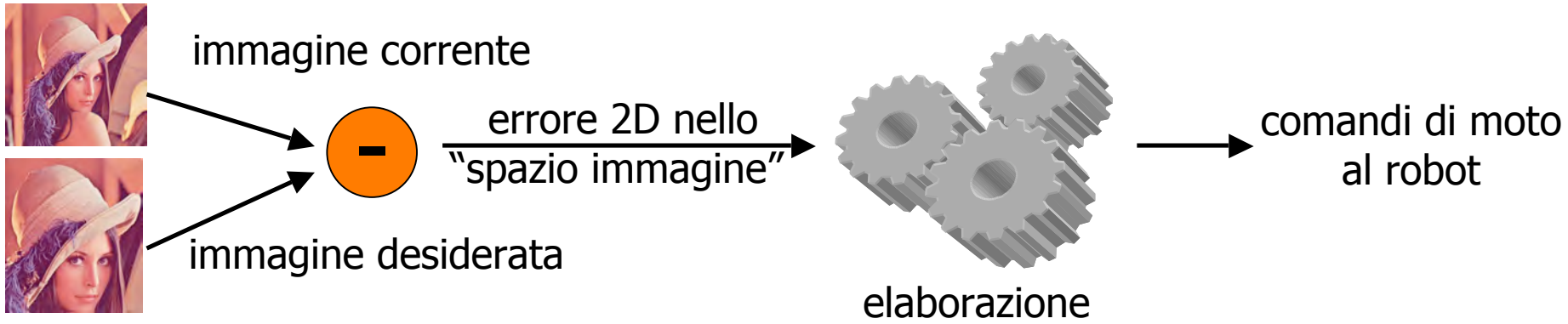
- position-based visual servoing (**PBVS**)
  - le informazioni estratte dalle immagini (features) vengono usate per ricostruire la posa 3D corrente dell'oggetto (posizione/orientamento)
  - combinandola con la conoscenza della posa 3D desiderata si genera un segnale di errore "cartesiano" che guida il robot verso il suo goal
- image-based visual servoing (**IBVS**)
  - l'errore viene calcolato direttamente sui valori numerici delle features estratte, senza passare per la ricostruzione 3D
  - il robot si muove in modo da portare le features correnti (quello che "vede" nella telecamera) verso i loro valori desiderati

Confronto tra i due schemi

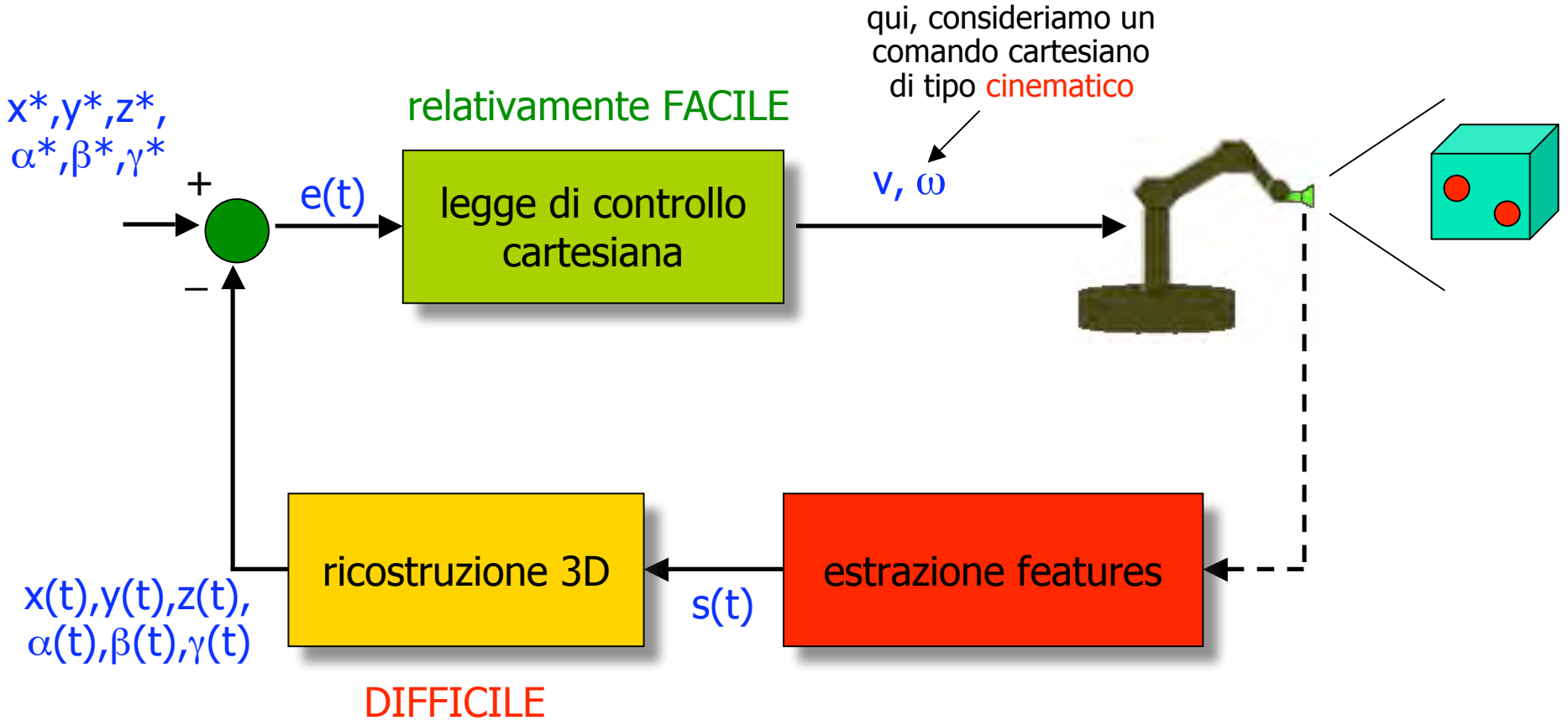
- position-based visual servoing (**PBVS**)



- image-based visual servoing (**IBVS**)

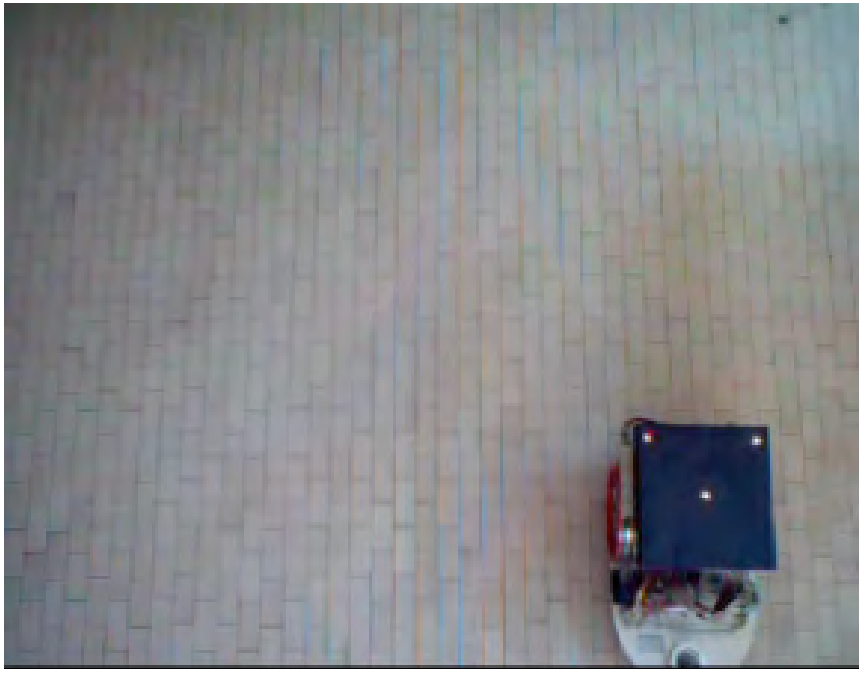


Architettura PBVS

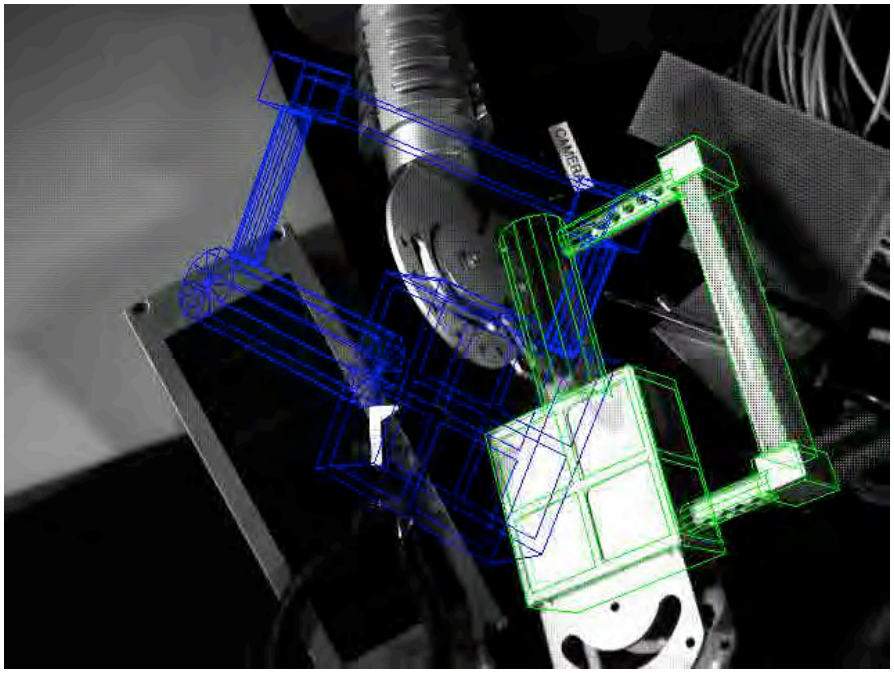


molto sensibile ai parametri di calibrazione della telecamera

Esempi di PBVS



eye-to-"robot" (SuperMario)



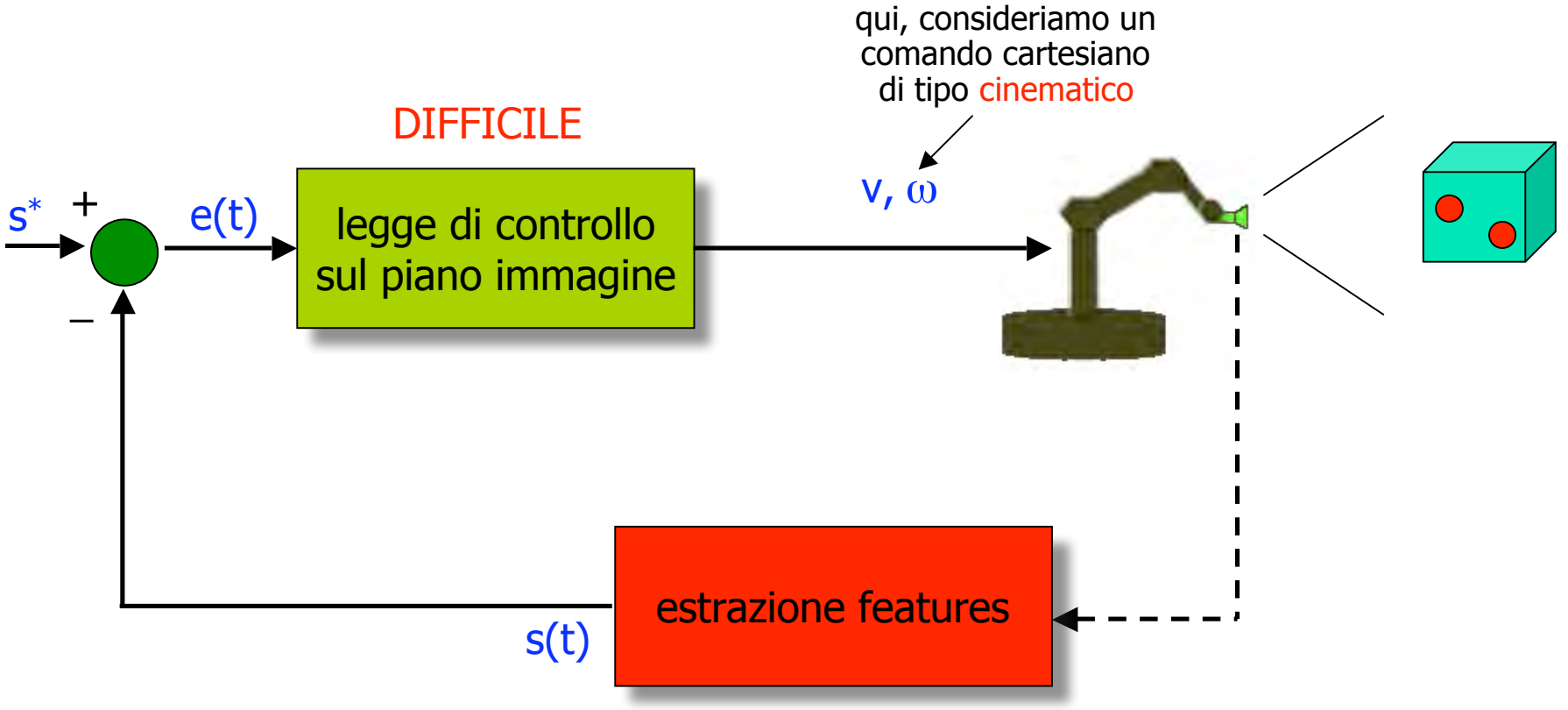
eye-in-hand (robot Puma)

posizione/orientamento della telecamera
geometria della scena

note a priori!

posizione e orientamento del robot
(a base mobile o fissa)

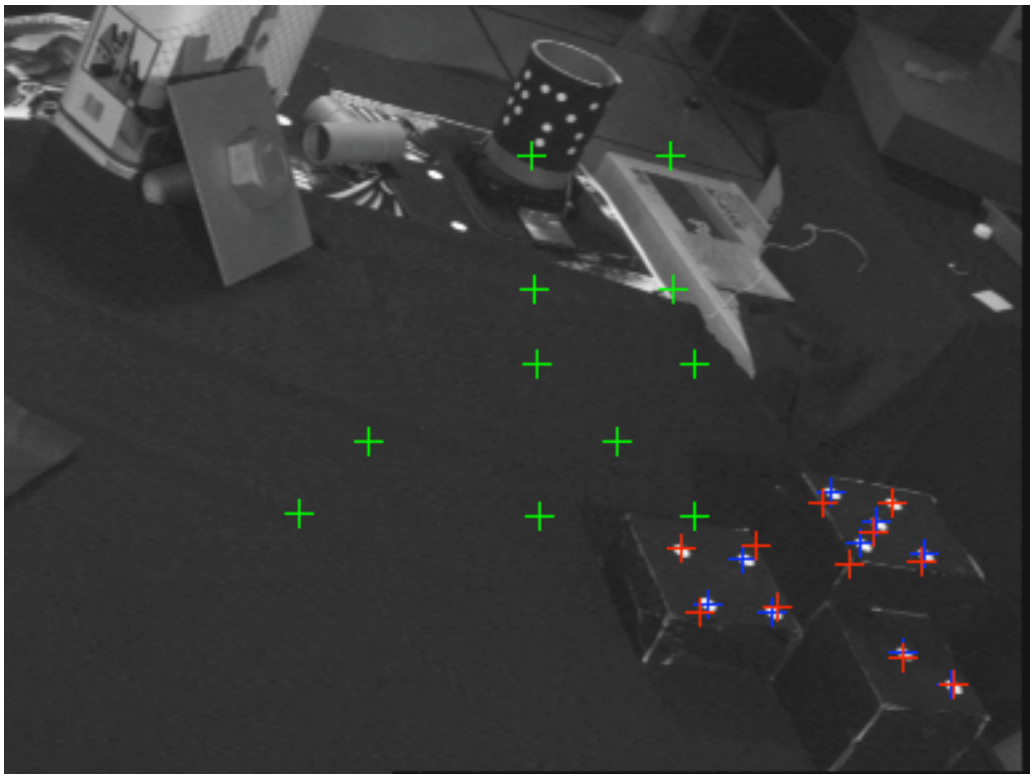
Architettura IBVS



quasi insensibile ai parametri intrinseci/calibrazione della telecamera

Un esempio di IBVS

qui le features sono punti (selezionati o in opportune combinazioni)



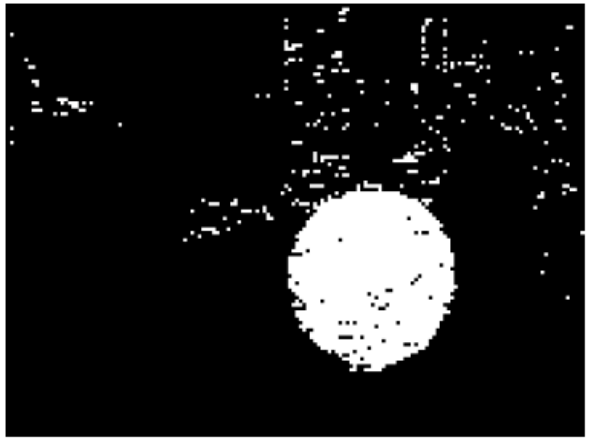
posizione desiderata features
posizione corrente features

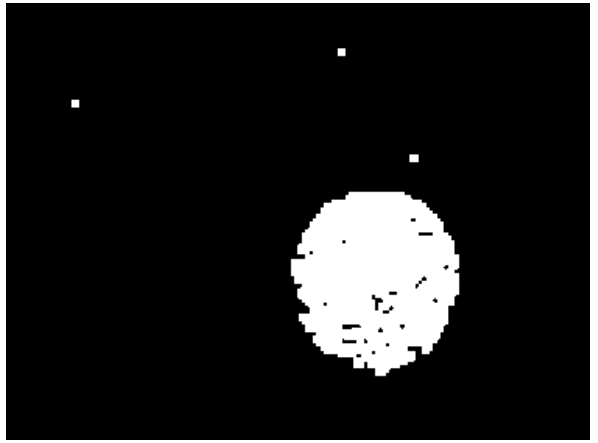
→ l'errore nello "spazio" delle features guida il controllo del moto del robot

Passi di uno schema IBVS

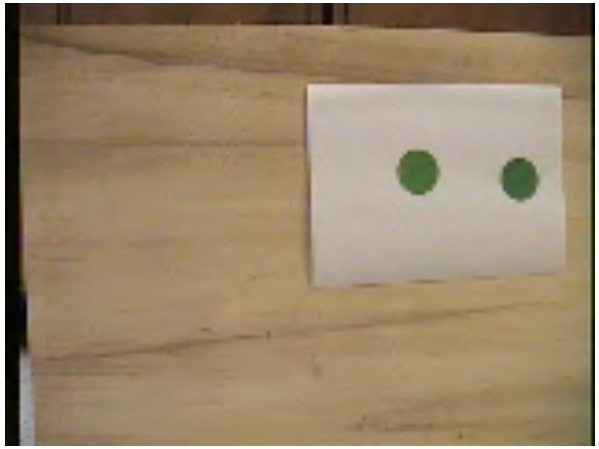
- acquisizione immagine
  - frame rate, ritardi, rumore, ...
- estrazione features
  - tecniche di Image Processing (può essere un passaggio molto impegnativo!)
- confronto con features "desiderate"
  - definizione del segnale d'errore nel piano immagine
- generazione del moto imposto alla telecamera
  - trasformazioni prospettiche
  - cinematica del manipolatore
  - legge di controllo cinematica (più spesso) o dinamica (ad es., PD + compensazione gravità - vedi libro di testo)

Tecniche di Image Processing





binarizzazione sullo spazio RGB → erosione e dilatazione

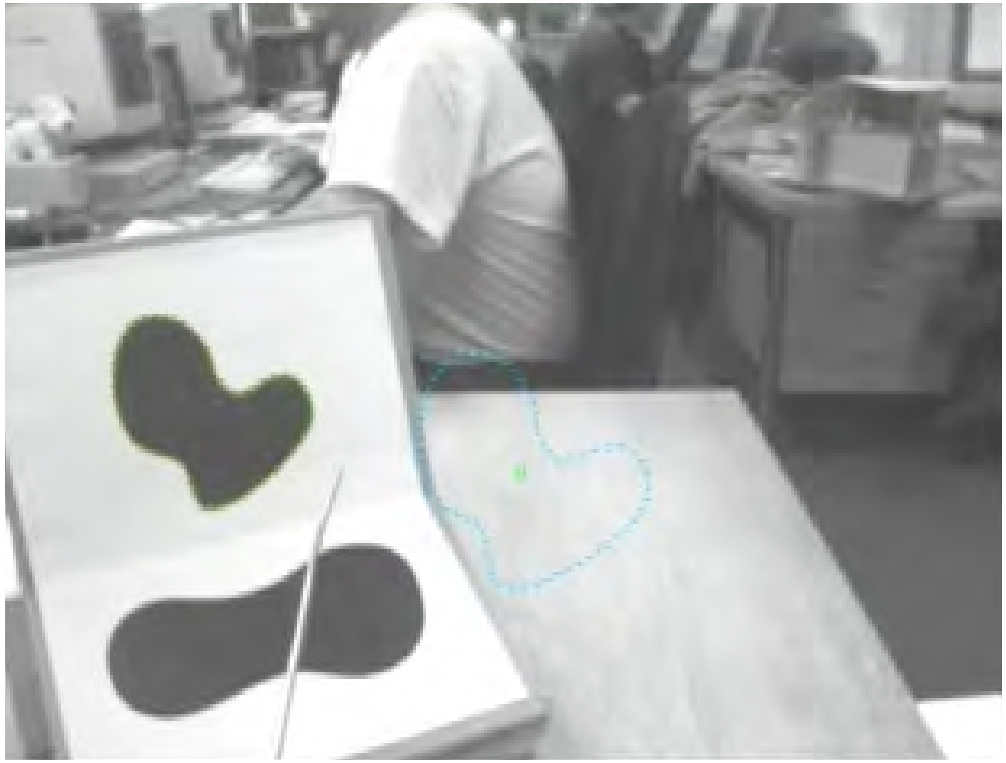


binarizzazione sullo spazio HSV → dilatazione

Cosa è una feature?

- image feature: qualsiasi caratteristica/struttura geometrica estratta dalla scena ripresa
  - punti
  - linee
  - ellissi (o qualsiasi altro contorno 2D)
- feature parameter(s): qualsiasi quantità numerica associata alla feature selezionata
  - coordinate di un punto
  - coefficiente angolare e offset di una linea
  - centro e raggio di una circonferenza
  - area e baricentro di un contorno 2D
  - "momenti" generalizzati dell'oggetto target nell'immagine
    - si possono costruire in modo che siano invarianti a scala e rotazione

Esempio di IBVS con momenti

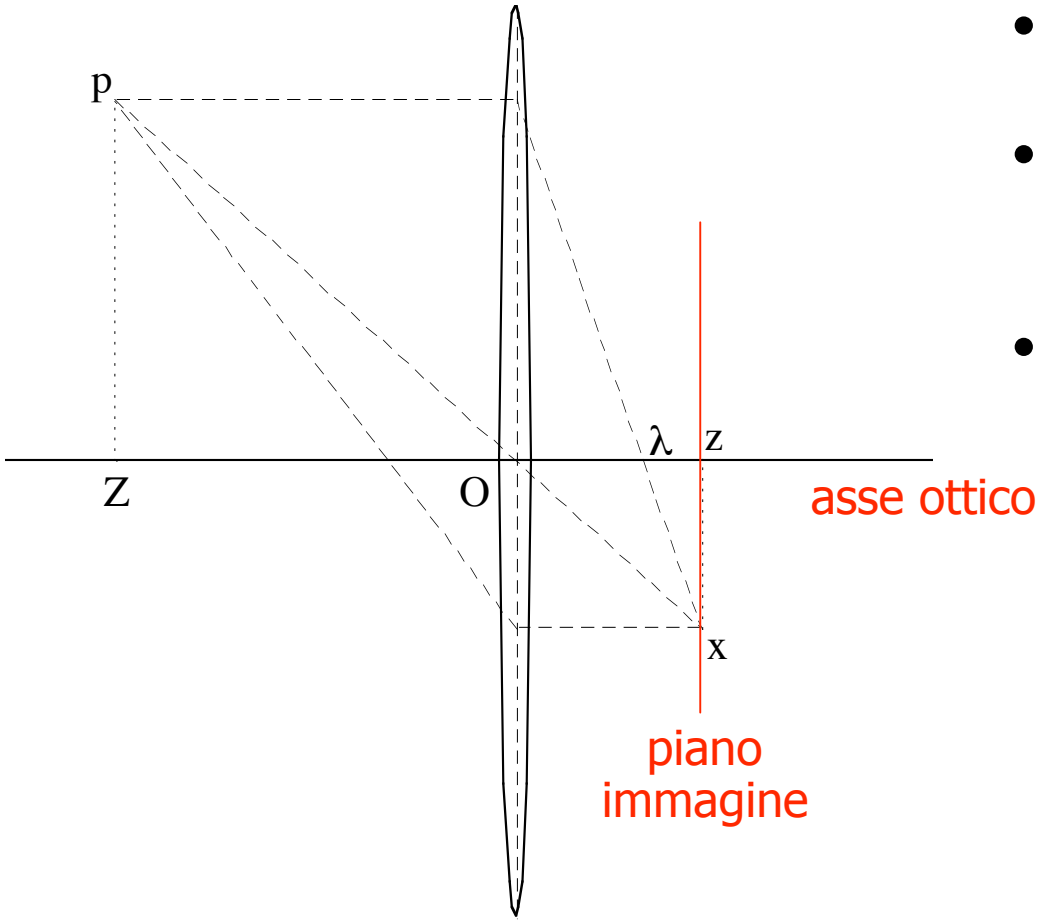


- evita il problema di individuare "corrispondenze" tra punti
- non è però facile controllare tutti i 6 dofs della telecamera utilizzando solo i momenti come features

Modello della telecamera

- set di lenti per indirizzare la luce entrante
  - lenti sottili (convergenti)
  - approssimazione pin-hole
  - lenti/sistemi catadiottrici (combinate con specchi)
- matrice di elementi sensibili alla luminosità (pixels nel piano immagine), eventualmente selettivi nei colori RGB
  ~ occhio umano
- frame grabber che "scatta delle istantanee" e le digitalizza in uscita
  - scheda + software su PC
  - frame rate = frequenza di uscita del nuovo "frame" digitalizzato
  - utile poter indirizzare un sottoinsieme (area) di pixel

Modello della lente sottile



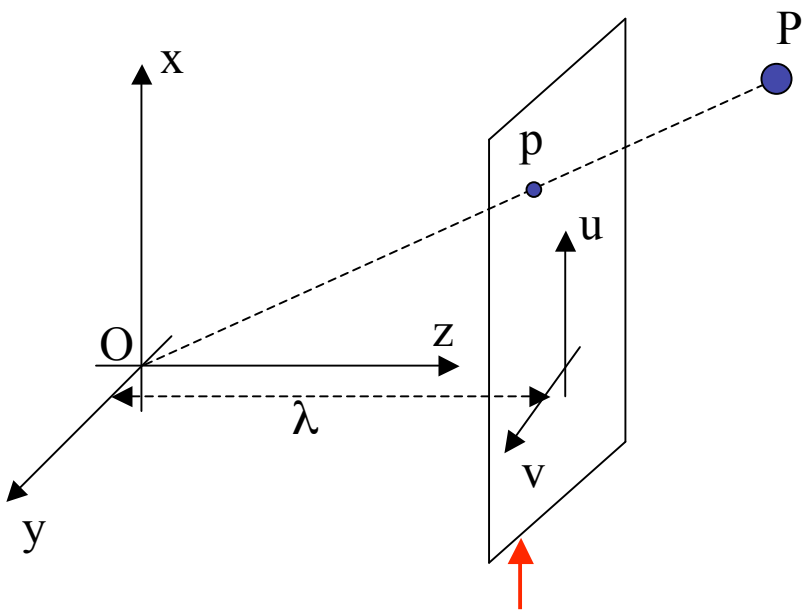
- ottica geometrica
- raggi paralleli all'asse ottico passano per λ (distanza focale)
- raggi per il centro ottico O non vengono deflessi

equazione fondamentale della lente sottile

$$\frac{1}{Z} + \frac{1}{z} = \frac{1}{\lambda}$$

potere diottrico

Modello pinhole



piano immagine dove sono posti
gli elementi sensibili alla luce
(in realtà speculare
rispetto al centro focale)

- apertura della lente sottile trascurabile
- rimangono solo i raggi passanti per il centro ottico
- dalle relazioni tra triangoli simili si ottengono le equazioni prospettiche

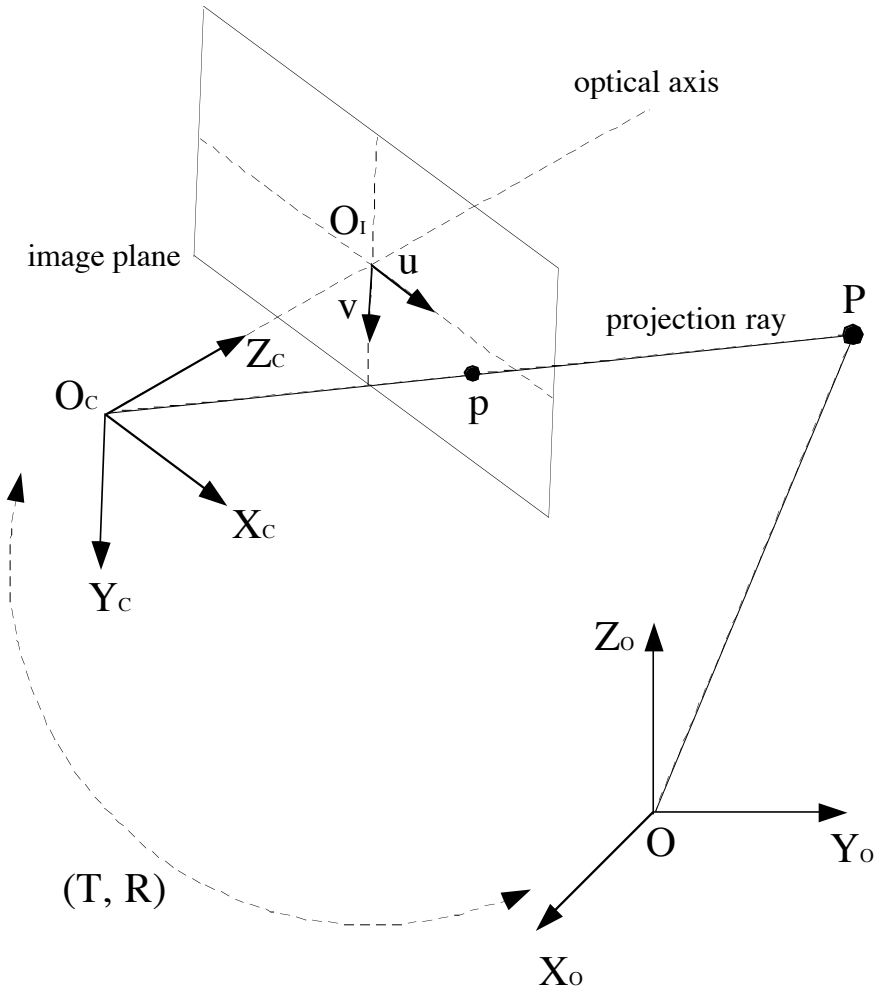
$$x = \lambda \frac{X}{Z}, \qquad y = \lambda \frac{Y}{Z}$$

con

$P = (X, Y, Z)$ punto cartesiano

$p = (x, y, \lambda)$ punto rappresentativo nel piano immagine

Terne di riferimento di interesse



- riferimento assoluto

$$\mathcal{F}_0 : \left\{ O,\ \vec{X}_0,\ \vec{Y}_0,\ \vec{Z}_0 \right\}$$

- riferimento telecamera

$$\mathcal{F}_C : \left\{ O_C,\ \vec{X}_C,\ \vec{Y}_C,\ \vec{Z}_C \right\}$$

- riferimento piano immagine

$$\mathcal{F}_I : \{ O_I,\ \vec{u},\ \vec{v} \}$$

- posizione/orientamento di $\mathcal{F}_C$ rispetto a $\mathcal{F}_0$

$$(T,\ R)$$

(traslazione, rotazione)

Matrice di interazione

- dato un set di features (parameters) $f = [f_1 \dots f_k]^T \in \mathbb{R}^k$
- siamo interessati al legame differenziale (cinematico) tra moto imposto alla telecamera e moto delle features sul piano immagine

$$\dot{f} = J(\cdot) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

- $(V, \Omega) \in \mathbb{R}^6$ rappresenta la velocità lineare/angolare della telecamera, espressa in $\mathcal{F}_C$

- $J(\cdot)$ è una matrice k*6, chiamata matrice di interazione
- consideriamo inizialmente solo point features, gli altri casi (aree, linee, …) sono estensioni di questa analisi

Calcolo della matrice di interazione

point feature, modello pinhole

- dalle equazioni prospettiche $u = \lambda \frac{X_C}{Z_C},\ v = \lambda \frac{Y_C}{Z_C}$ si ottiene

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \frac{\lambda}{Z} & 0 & -\frac{u}{Z} \\ 0 & \frac{\lambda}{Z} & -\frac{v}{Z} \end{bmatrix} \begin{bmatrix} \dot{X}_C \\ \dot{Y}_C \\ \dot{Z}_C \end{bmatrix} = J_1(u, v, \lambda) \begin{bmatrix} \dot{X}_C \\ \dot{Y}_C \\ \dot{Z}_C \end{bmatrix}$$

- la velocità $\left(\dot{X}_C, \dot{Y}_C, \dot{Z}_C\right)$ del punto P nel riferimento $\mathcal{F}_C$ è in realtà dovuta alla roto-traslazione (V, Ω) della telecamera (P è fermo in $\mathcal{F}_0$)

- legame cinematico tra $\left(\dot{X}_C, \dot{Y}_C, \dot{Z}_C\right)$ e (V, Ω)

$$\begin{bmatrix} \dot{X}_C \\ \dot{Y}_C \\ \dot{Z}_C \end{bmatrix} = -V - \Omega \times \begin{bmatrix} X_C \\ Y_C \\ Z_C \end{bmatrix}$$

Calcolo della matrice di interazione (cont)
point feature, modello pinhole

- l'ultima equazione si può esprimere in forma matriciale

$$\begin{bmatrix} \dot{X}_C \\ \dot{Y}_C \\ \dot{Z}_C \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & -Z_C & Y_C \\ 0 & -1 & 0 & Z_C & 0 & -X_C \\ 0 & 0 & -1 & -Y_C & X_C & 0 \end{bmatrix} \begin{bmatrix} V \\ \Omega \end{bmatrix} = J_2(X_C, Y_C, Z_C) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

- combinando, si ha la matrice di interazione per una point feature

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = J_1 J_2 \begin{bmatrix} V \\ \Omega \end{bmatrix} = \begin{bmatrix} -\frac{\lambda}{Z} & 0 & \frac{u}{Z} & \frac{uv}{\lambda} & -\left(\lambda + \frac{u^2}{\lambda}\right) & v \\ 0 & -\frac{\lambda}{Z} & \frac{v}{Z} & \lambda + \frac{v^2}{\lambda} & -\frac{uv}{\lambda} & -u \end{bmatrix} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

$$= J_p(u, v, Z) \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

p = point (feature)

Commenti

- la interaction matrix nella relazione

$$\begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \boxed{J_p(u, v, Z)} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

  - dipende dai valori correnti della feature e dalla sua depth Z

  - $\dim \ker J_p = 4$, quindi esistono ∞^4 movimenti della telecamera che sono inosservabili sul piano immagine
    - ad esempio, una traslazione lungo il raggio di proiezione

  - nel caso di più point features considerate, le relative matrici si "impilano" una sull'altra
    - p point features: matrice di interazione k × 6, con k = 2p

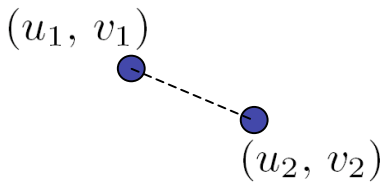
Altri esempi di matrici di interazione

- distanza tra due point features

$$d = \sqrt{(u_1 - u_2)^2 + (v_1 - v_2)^2}$$

$$\dot{d} = \frac{1}{d} \begin{bmatrix} u_1 - u_2 & v_1 - v_2 & u_2 - u_1 & v_2 - v_1 \end{bmatrix} \begin{bmatrix} \dot{u}_1 \\ \dot{v}_1 \\ \dot{u}_2 \\ \dot{v}_2 \end{bmatrix}$$

$$= J_d(u_1, u_2, v_1, v_2) \begin{bmatrix} J_{p1}(u_1, v_1, Z_1) \\ J_{p2}(u_2, v_2, Z_2) \end{bmatrix} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$



- image moments

$$m_{ij} = \int\int_{\mathcal{R}(t)} x^i y^j \, dx dy$$

$\mathcal{R}(t)$ porzione del piano immagine occupata dall'oggetto

  - utili per rappresentare informazioni qualitative (area, baricentro, orientamento dell'ellisse approssimante, …)
  - sfruttando le formule di Green e la matrice di interazione di una generica point feature si ottiene

$$\dot{m}_{ij} = L_{ij} \begin{bmatrix} V \\ \Omega \end{bmatrix}$$

Cinematica differenziale del robot

- caso eye-in-hand: telecamera posizionata sull'end-effector di un braccio manipolatore (a base fissa o montato su base mobile)
- il moto imposto alla telecamera (V, Ω) coincide con la velocità lineare/angolare dell'end-effector e viene realizzato dalle velocità di giunto del manipolatore (o più in generale dai comandi di velocità ammissibili di un manipolatore mobile)

$$\begin{bmatrix} V \\ \Omega \end{bmatrix} = J_m(q)\dot{q} = J_m(q)u$$

ingresso di controllo in velocità

Jacobiano geometrico del manipolatore

eventuale Jacobiano NMM del manipolatore mobile

con $q \in \mathbb{R}^n$ vettore delle variabili di configurazione del robot

- in generale, in questo Jacobiano occorre una calibrazione hand-eye

Image Jacobian

- combinando i passaggi dalla velocità delle point features sul piano immagine alla velocità dei giunti/comando ammissibile di velocità del robot, si ha

$$\dot{f} = J_p(f, Z)J_m(q)u = J(f, Z, q)u$$

- la matrice $J(f, Z, q)$ è l'Image Jacobian e fa le veci del classico Jacobiano del robot
- si possono quindi applicare le tecniche classiche di controllo cinematico (o anche dinamico) del moto dei robot
- la variabile di uscita (controllata) è il vettore delle features nel piano immagine (spazio del compito!)
- in questo spazio è dunque definito anche l'errore del compito

Controllo cinematico per IBVS

- definito il vettore di errore $e = f_d - f(t)$, la scelta generale

$$u = J^{\dagger}(\dot{f}_d + ke) + (I - J^{\dagger}J)u_0$$

soluzione a norma minima — termine in $\ker J$, non "muove" le features

azzera esponenzialmente l'errore sul compito (a meno di singolarità, fondo corsa, scomparsa delle features dal piano immagine)

- l'errore, ovvero il segnale di feedback, dipende solo dalle features!
- il vettore u_0, nel caso ridondante, può essere usato per ottimizzare altri criteri
- rimane la dipendenza di J dalla depth Z delle features
  - uso del valore costante e "noto" nella posa desiderata $J(f,\ Z^*,\ q)$
  - stima in linea del valore corrente mediante un osservatore

Esempio

- base mobile (uniciclo) + manipolatore 3R
- configurazione eye-in-hand

- 5 comandi
  - velocità lineare e angolare base mobile
  - velocità dei tre giunti del manipolatore

- task specificato da 2 point features ⟶ 4 variabili

$$f = [f_{u1} \; f_{v1} \; f_{u2} \; f_{v2}]^T$$

- 5 - 4 = 1 grado di ridondanza

Simulazione

vista dalla telecamera

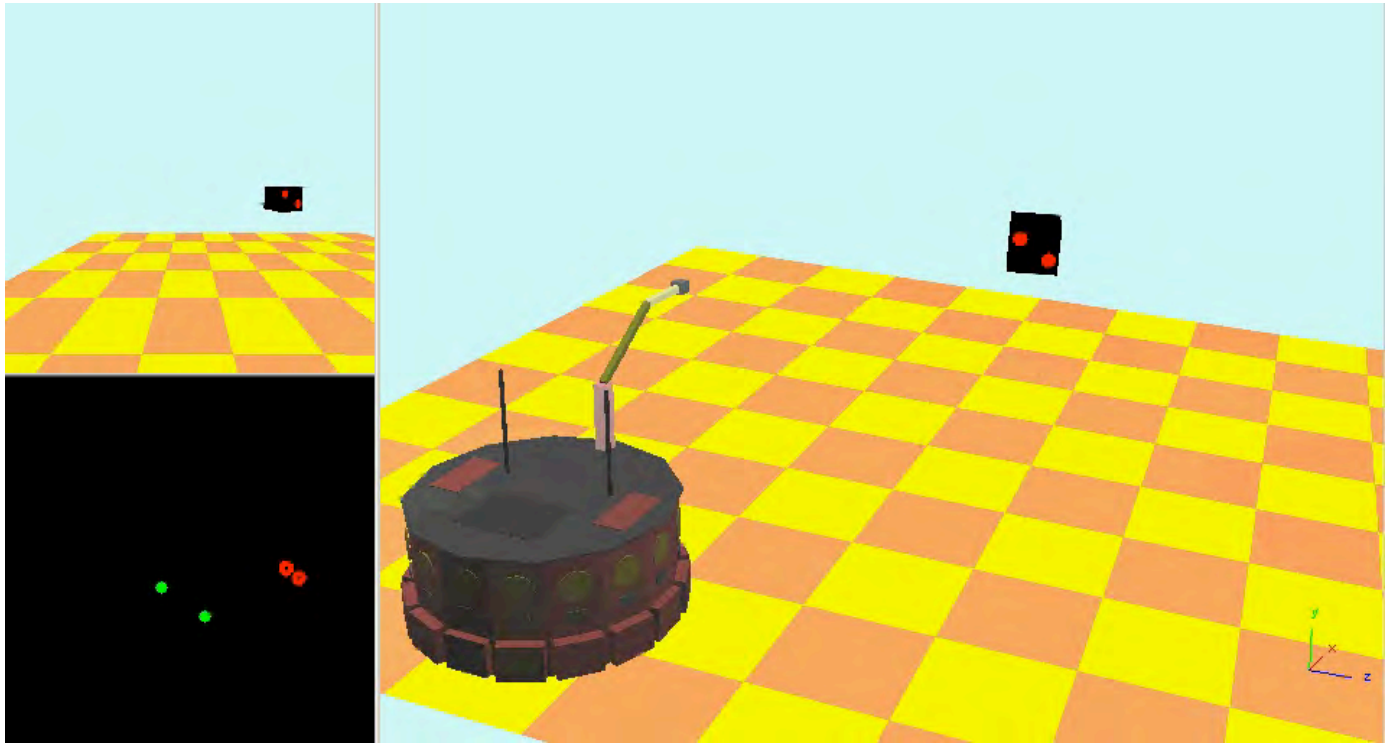
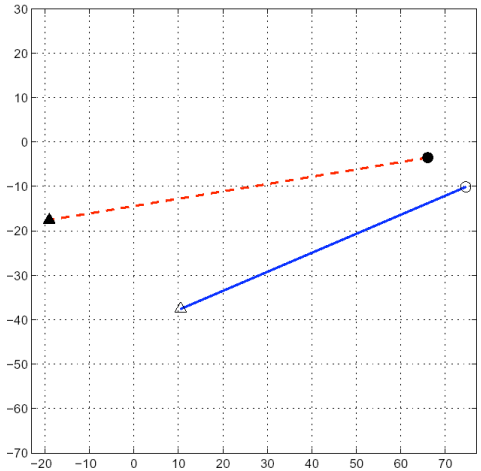


immagine elaborata



andamento delle features

- simulazione in Webots
- valore corrente Z supposto noto
- k diagonale (disaccoppiamento!)
- "traiettorie rettilinee" nel piano immagine

Controllo IBVS con Task Sequencing

- approccio a fasi: spostare una feature per volta tenendo "ferme" le altre, sfruttando i movimenti inosservabili sul piano immagine
- Image Jacobians delle singole features

$$\dot{f}_1 = J_1 u, \; \dot{f}_2 = J_2 u$$

- la prima feature f_1 viene regolata nella prima fase con

$$u = J_1^{\dagger} k_1 e_1 + (I - J_1^{\dagger} J_1) u_0$$

- la feature f_2 si fa muovere nella seconda fase nel "nullo" della prima

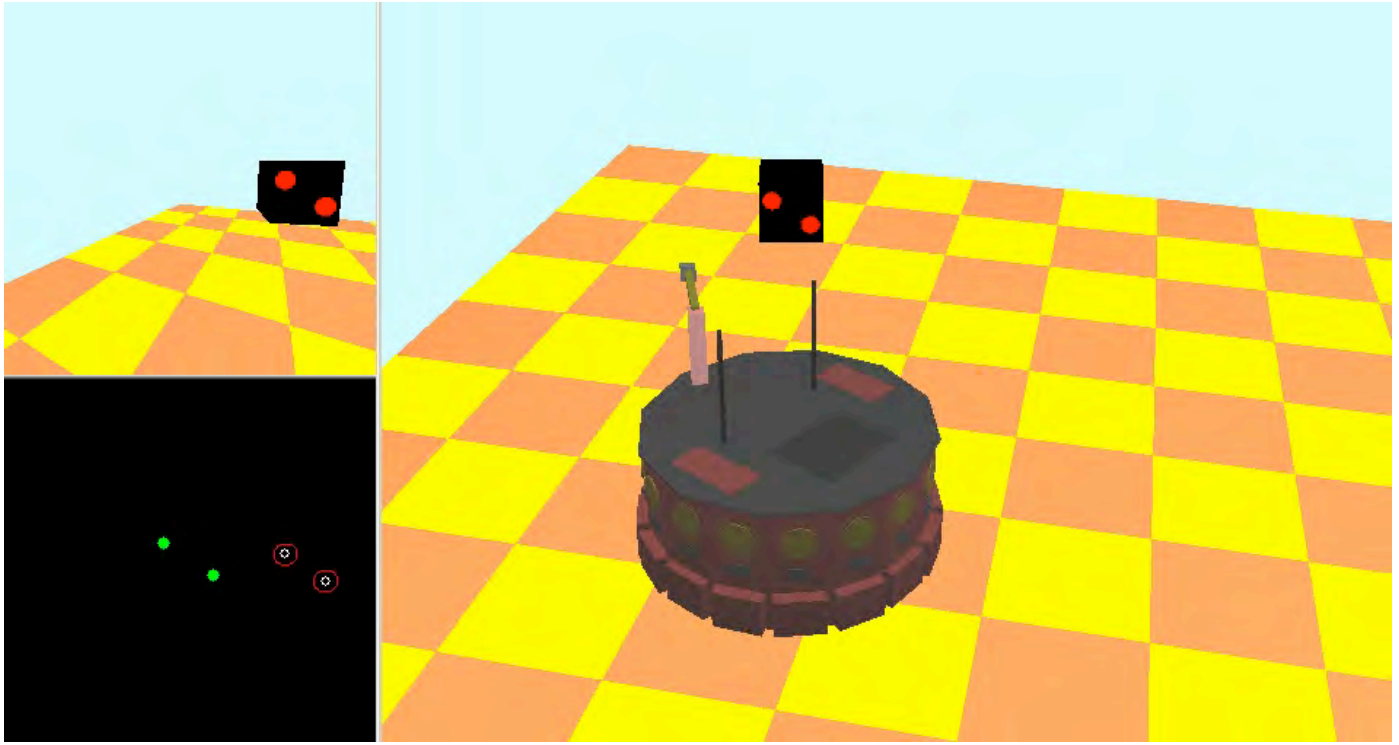
$$u = (I - J_1^{\dagger} J_1) J_2^T k_2 e_2$$

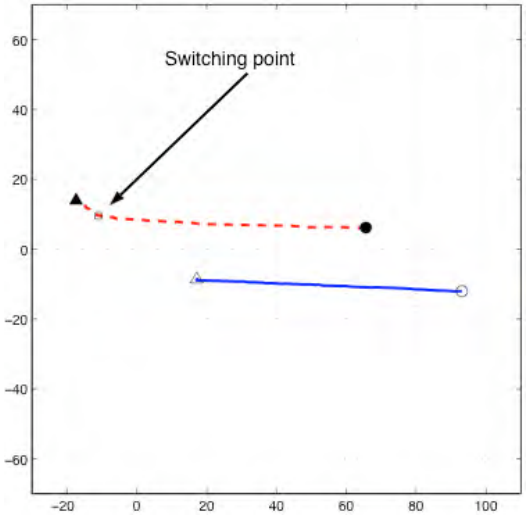
- durante la prima fase ho due gradi di ridondanza aggiuntivi rispetto al caso "completo", che si possono usare ad es. per migliorare l'allineamento del robot rispetto al target
- nel caso di uniciclo + manipolatore 2R, il caso completo NON avrebbe ridondanza e senza il sequenziamento incontrerei singolarità…

Simulazione con TS

base mobile (uniciclo) + manipolatore polare 2R (Image Jacobian "quadrato")

point feature 1
(regolata nella
prima fase)
point feature 2
(nella seconda fase)



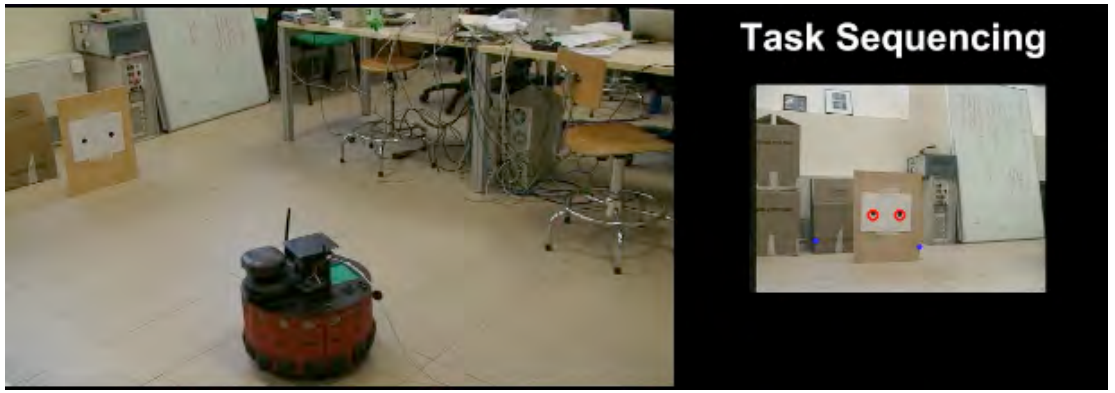


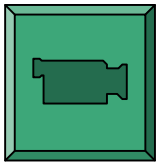
andamento
delle features

- simulazione in Webots
- valore corrente Z supposto noto

Esperimenti

Magellan (uniciclo) + camera pan-tilt (mobilità simile al robot polare 2R)





video
IEEE ICRA'08

- confronto tra schema con Task Priority (TP) e con Task Sequencing (TS)
- entrambi gestiscono le eventuali singolarità incontrate (Image Jacobian)
- frame rate della telecamera = 7 Hz

Nelle situazioni reali...

- in pratica si ha incertezza su una serie di dati
  - distanza focale λ (parametri intrinsici della telecamera)
  - calibrazione hand-eye (parametri estrinseci della telecamera)
  - depth (profondità) Z delle point features
  - ....
- si può calcolare solo un'approssimazione dell'Image Jacobian (sia nella sua parte di interaction matrix, che nello Jacobiano del robot)
- ad anello chiuso, la dinamica di errore sulle features è $\dot{e} = -J\widehat{J}^{\dagger}Ke$
  - caso ideale: $JJ^{\dagger} = I$ caso reale: $J\widehat{J}^{\dagger} \neq I$
- si può dimostrare che una condizione sufficiente di convergenza locale dell'errore a zero (il sistema d'errore è lineare, tempo-variante) è

$$J\widehat{J}^{\dagger} > 0$$

Un osservatore della depth Z

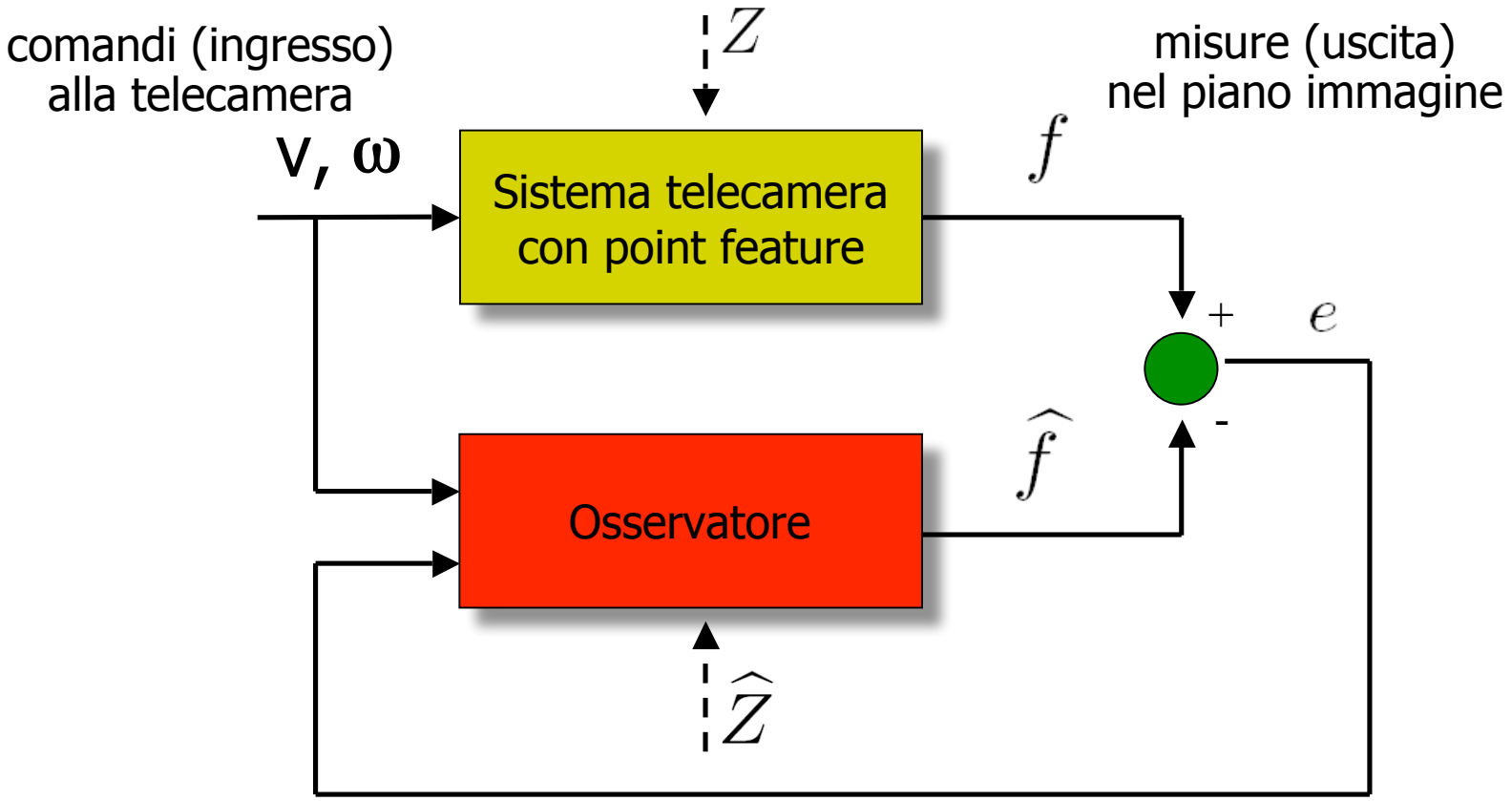
- si può stimare in linea il valore corrente (variabile) della depth Z, per ciascuna point feature considerata, con un osservatore
- definiti $x = [u,\ v,\ 1/Z]^T$, $\widehat{x} = [\widehat{u},\ \widehat{v},\ 1/\widehat{Z}]^T$ come stato corrente e stato stimato e $y = [u,\ v]^T$ come uscita misurata
- un osservatore (non lineare) di x

$$\dot{\widehat{x}} = \alpha(\widehat{x},\ y)u + \beta(\widehat{x},\ y,\ u)$$

garantisce $\lim_{t\to\infty} ||x(t) - \widehat{x}(t)|| = 0$ se

  - la velocità lineare della telecamera non è nulla
  - il vettore di velocità lineare non è allineato con il raggio di proiezione della point feature considerata
  - sono condizioni di "persistent excitation" (~ condizioni di osservabilità)

Schema a blocchi dell'osservatore



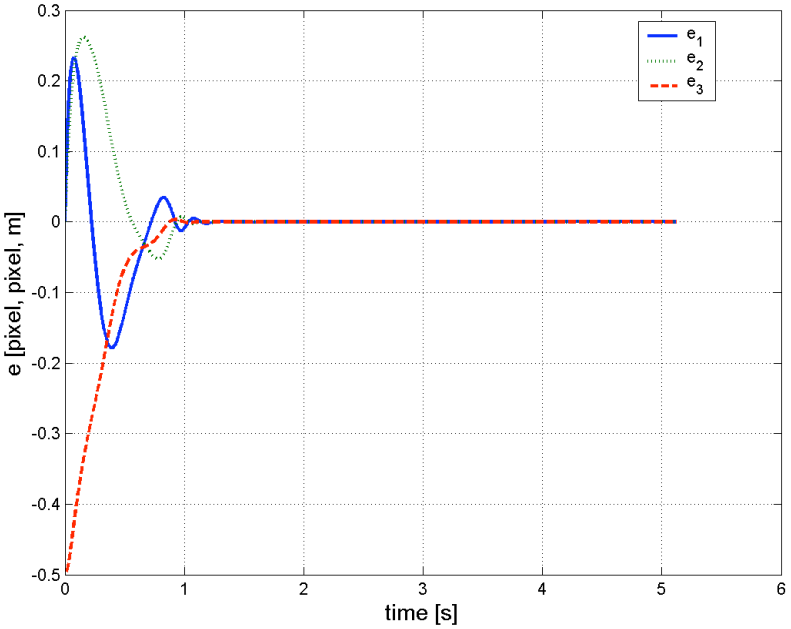
Risultati di osservazione della Z

stato reale e stimato iniziali

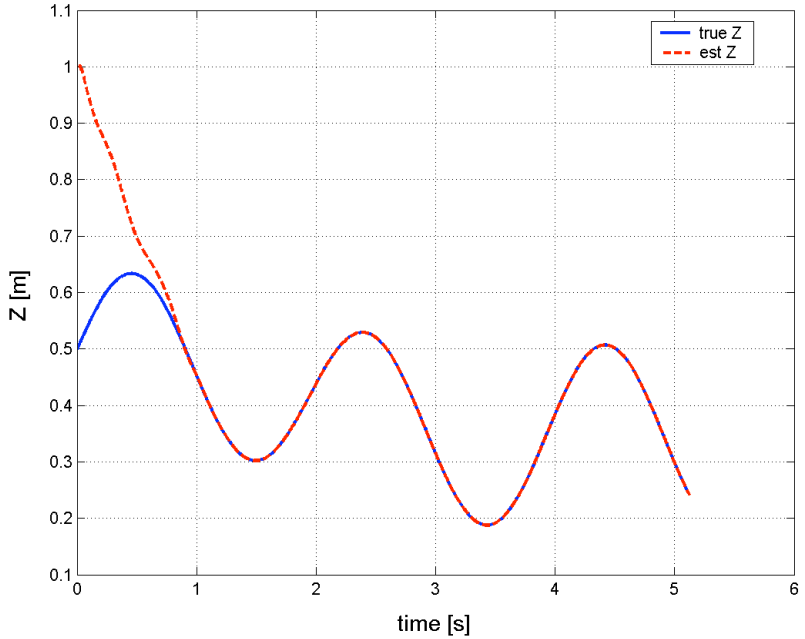
$$x(t_0) = [10 \quad -10 \quad 2]^T$$
$$\widehat{x}(t_0) = [10 \quad -10 \quad 1]^T$$

$$v_x(t) = 0.1 \cos 2\pi t$$
$$v_z(t) = 0.5 \cos \pi t$$
$$\omega_x(t) = 0.6 \cos \pi/2\, t$$
$$\omega_z(t) = 1$$

comandi ad anello aperto



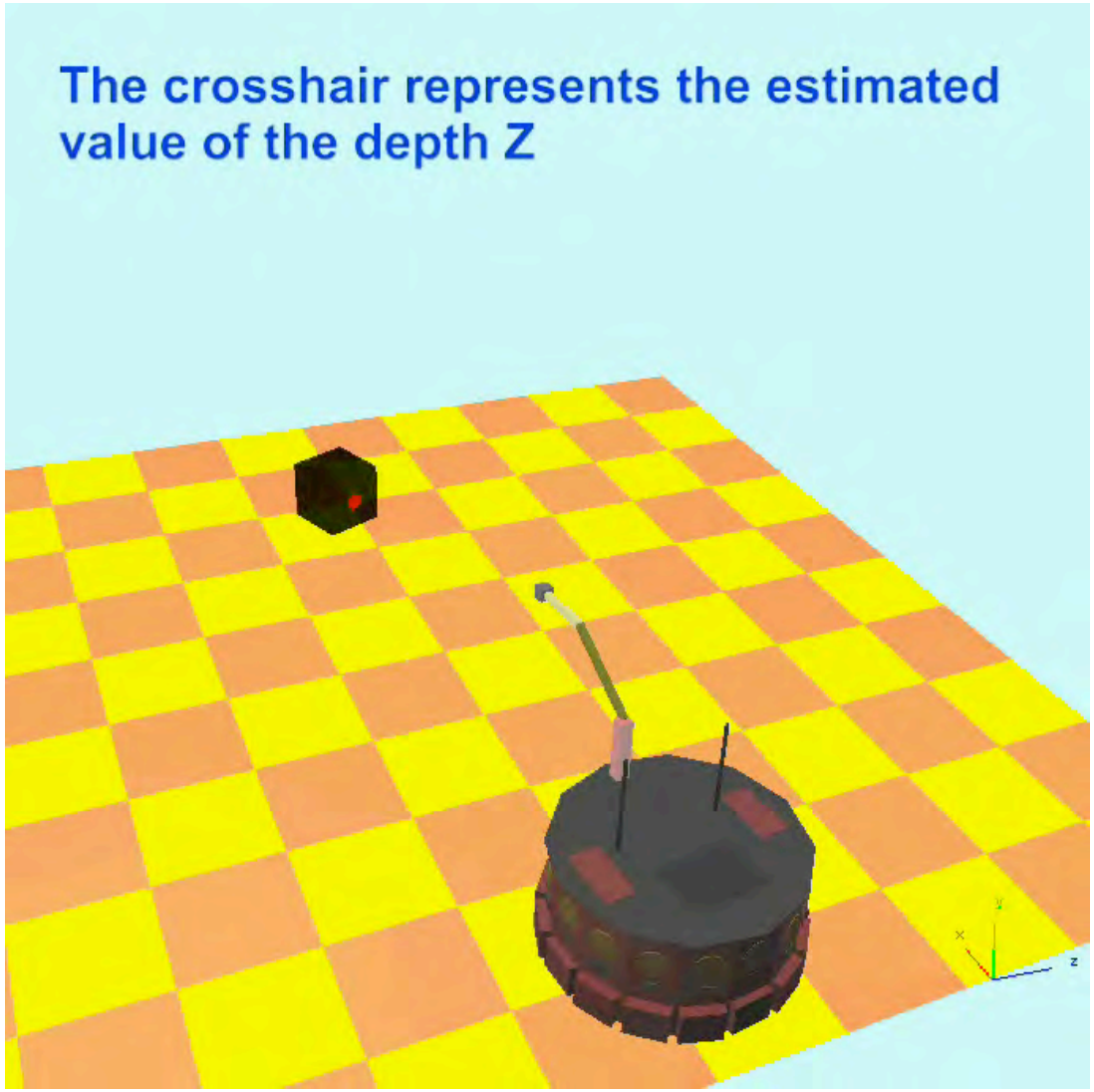
evoluzione errore di stima $e(t)$



$Z(t)$ e $\widehat{Z}(t)$ nel tempo

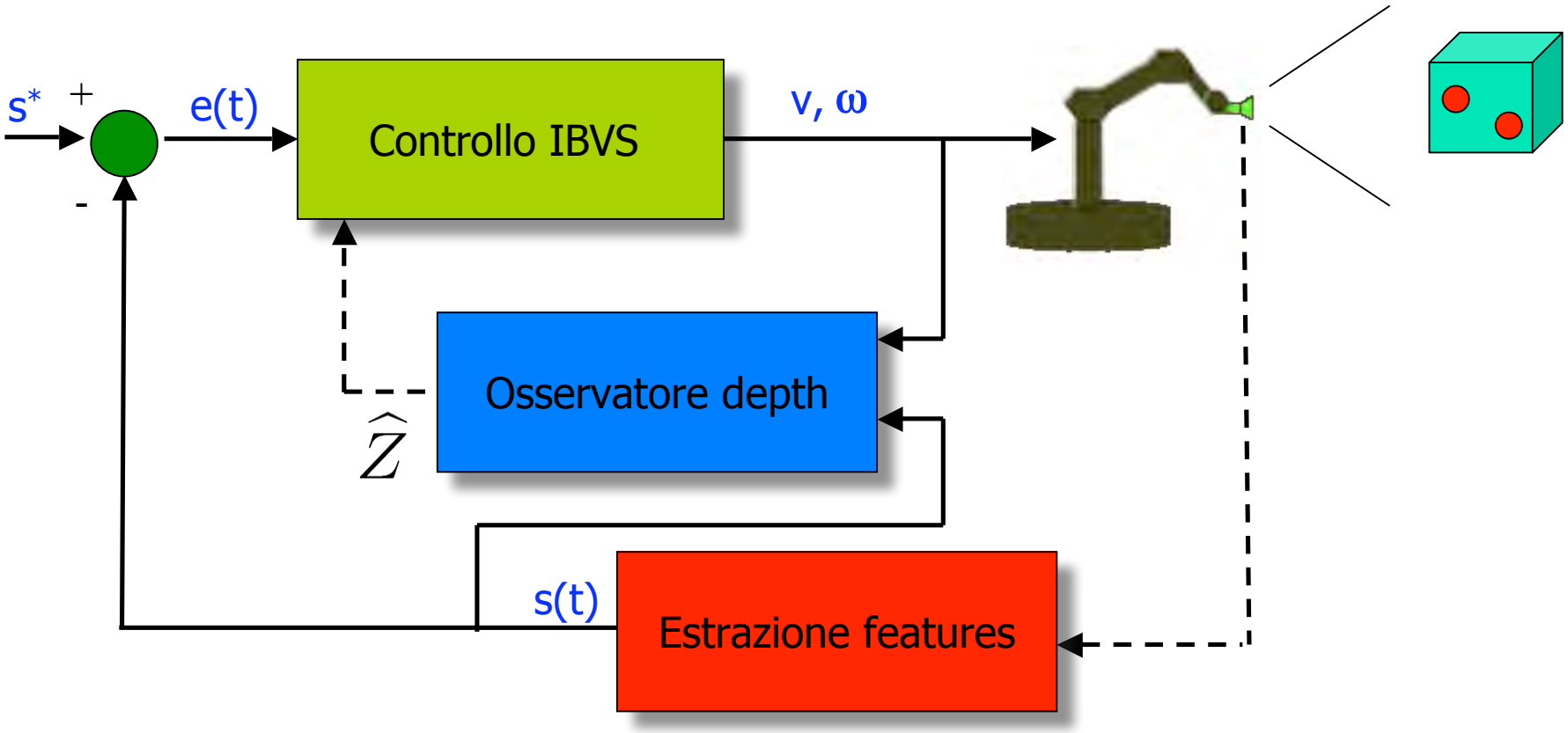
Simulazione dell'osservatore

con spostamento del target



Controllo IBVS con osservatore

- l'osservatore della depth può essere facilmente inserito in linea con un qualunque schema di controllo IBVS



Set-up sperimentale

- Visual Servoing con telecamera fissa su robot mobile skid-steering
- calibrazione eye-robot molto rozza
- depth dei target points incognite (stimate in linea)

Esperimenti

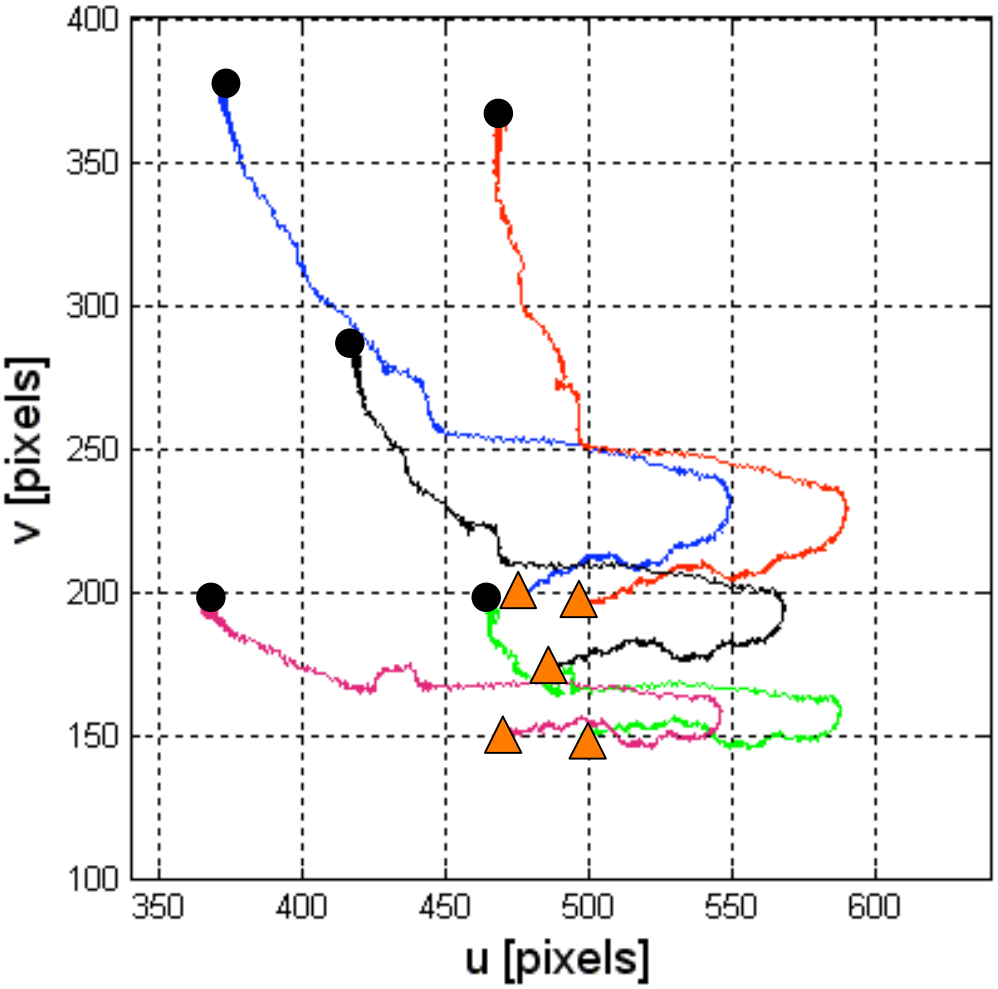
- il moto delle features sul piano immagine non è perfetto
- il compito di posizionamento visuale è comunque realizzato

vista dall'esterno

vista dalla telecamera

c/o Università di Siena

Evoluzione reale delle features



- movimento delle 5 features nel piano immagine
- verso la fine il movimento è ≈ in linea retta, perchè l'osservatore della depth è andato a convergenza
- l'Image Jacobian calcolato è prossimo a quello reale

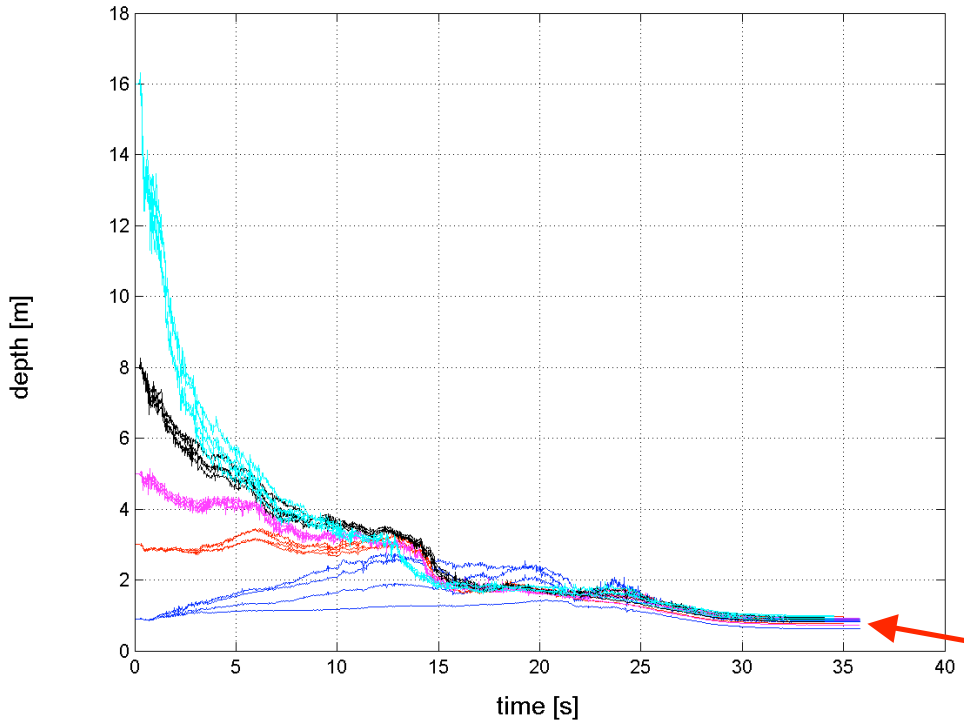
$$\widehat{Z} \rightarrow Z \Rightarrow \widehat{J} \rightarrow J$$

- depths reali iniziali e finali

$$Z(t_0) \simeq 4\,m \quad Z_d \simeq 0.9\,m$$

Esperimenti con l'osservatore

- è stato eseguito lo stesso compito con 5 inizializzazioni differenti per l'osservatore delle depth, variandole tra 0.9 m (= depth nella posa finale desiderata) e 16 m (molto maggiore di quella iniziale effettiva)



evoluzioni delle depth stimate

- valori iniziali della stima delle depth nell'osservatore

$$\begin{cases} \hat{Z}_1(t_0) = 16\,m \\ \hat{Z}_2(t_0) = 8\,m \\ \hat{Z}_3(t_0) = 5\,m \\ \hat{Z}_4(t_0) = 3\,m \\ \hat{Z}_5(t_0) = 0.9\,m \end{cases}$$

- depth reale nella posa iniziale

$$Z(t_0) \simeq 4\,m$$

- depth reale nella posa finale

$$Z_d \simeq 0.9\,m$$