

# Realizzazione di un software per la stima del posizionamento di un corpo rigido con metodi di visione e validazione attraverso l'uso di un motore grafico

Luca Calacci

Università degli Studi di Roma - Tor Vergata

*luca.calacci@gmail.com*

5 ottobre 2017

# Contenuti

1

## Introduzione

2

## Stima della matrice di trasformazione

- Il metodo di Groebner
- Il metodo di Kabsch

3

## Visione artificiale

- Features detection
- Stereovisione: calcolo delle coordinate 3D

4

## Implementazione

- Analisi dei requisiti
- L'architettura

5

## Test delle prestazioni

6

## Conclusioni e sviluppi futuri

## La richiesta

Siano dati diversi sensori, solidali ad un corpo rigido, e le relative misure di velocità angolare, si richiede, se possibile, la stima della matrice di rotazione tra essi relativa.

# Il punto di partenza

## La richiesta

Siano dati diversi sensori, solidali ad un corpo rigido, e le relative misure di velocità angolare, si richiede, se possibile, la stima della matrice di rotazione tra essi relativa.

## Generalizzazione del problema

- Velocità angolare  $\rightarrow$  Vettore in  $\mathbb{R}^3$ ,
- Matrice di rotazione  $R \rightarrow$  Matrice di trasformazione  $T(R, t)$ 
  - $R :=$  matrice di rotazione,
  - $t :=$  vettore di traslazione,
- Si può stimare BIAS relativo tra i sensori.

## Cosa?

progettazione software di stima della posizione e orientamento mediante impiego di strumenti di visione.

## Perché?

- applicazione pratica degli strumenti di stima presentati,
- problema di rilievo in ambito robotico,
- studio di strumenti innovativi.

## Come?

- strumenti di **Visione Artificiale** per la ricerca, e il tracciamento di punti chiave in uno scenario ignoto.
  - Si scelgono dei punti "particolari" della scena e si effettua la loro ricerca nell'istante successivo (dopo lo spostamento)
  - Si ottiene un set di punti omologhi e se ne calcolano le coordinate
- Si effettua la stima della trasformazione a partire dal set ottenuto.

## Superimposizione di Procrustes

*dati due corpi ( $N$ -dimensionali), attraverso solo operazioni di rotazione, traslazione e ridimensionamento, si deve riuscire a far combaciare il primo sul secondo in modo ottimo. L'ottimo viene valutato attraverso una quantità detta distanza di Procrustes.*

Si presentano due soluzioni basate su approcci diversi:

- **Metodo di Groebner:** basato su strumenti di geometria algebrica: basi di Groebner, teoria dell'eliminazione.
- **Metodo di Kabsch:** metodo adattativo, robusto ai disturbi in misura.

## Il metodo di Groebner: Il caso planare

- Siano  $w_1, w_1' \in \mathbb{K}[w]^2$ ,  $t \in \mathbb{K}[t]^2$  e  $R \in \mathbb{K}[r]^{2 \times 2}$ ,
- si definisce l'ideale base attraverso il seguente set di polinomi:

$$\begin{bmatrix} p_1 \\ p_2 \end{bmatrix} := w_1 - R w_1' - t,$$

$$\begin{bmatrix} p_3 \\ p_4 \end{bmatrix} := w_2 - R w_2' - t,$$

$$p_5 := \det(R) - 1,$$

$$p_6 := (w_{2x} - w_{1x})^2 + (w_{2y} - w_{1y})^2 - (w_{2x}' - w_{1x}')^2 + \\ - (w_{2y}' - w_{1y}')^2$$

## Il metodo di Groebner: Il caso planare

- dove:  $R := \begin{bmatrix} r_{11}, & r_{12} \\ -r_{12}, & r_{11} \end{bmatrix}$ ,  $t := \begin{bmatrix} t_x \\ t_y \end{bmatrix}$ ,  $w_i := \begin{bmatrix} w_{ix} \\ w_{iy} \end{bmatrix}$
- Si definisce l'ideale  $I$  attraverso i polinomi generatori  $p_i$

$$I := \langle p_1, \dots, p_6 \rangle \quad (1)$$

- Si fissa l'ordinamento lessico-grafico seguente:  
 $r_{11} \geq_{Lex} r_{12} \geq_{Lex} t_x \geq_{Lex} t_y$
- Utilizzando il software Macaulay si calcola una base di Groebner  $G$  per l'ideale  $I$ :

$$G := \langle g_1, \dots, g_6 \rangle \quad (2)$$

# Il metodo di Groebner: Il caso planare

- dove i polinomi indicati hanno la seguente forma:

$$\begin{aligned}g_1 &= w_{2x}^{'2} - 2w_{1x}'w_{2x}' - w_{1x}^2 - w_{1y}^2 \\&\quad + w_{1x}^{'2} + 2w_{1x}w_{2x} - w_{2x}^2 + 2w_{1y}w_{2y} - w_{2y}^2, \\g_2 &= w_{2y}^{'2} - 2w_{1y}'w_{2y}' + w_{1y}^{'2} \\g_3 &= t_y + \phi_1(w_1, w_1', w_2, w_2')/\phi_d(w_1, w_1', w_2, w_2'), \\g_4 &= t_x - \phi_2(w_1, w_1', w_2, w_2')/\phi_d(w_1, w_1', w_2, w_2'), \\g_5 &= r_{12} - \phi_3(w_1, w_1', w_2, w_2')/\phi_d(w_1, w_1', w_2, w_2'), \\g_6 &= r_{11} - \phi_4(w_1, w_1', w_2, w_2')/\phi_d(w_1, w_1', w_2, w_2').\end{aligned}$$

- la funzione  $\phi_d(\cdot)$  non si annulla mai per  $w_1 \neq w_2$

## Il metodo di Groebner: Il caso planare

- La soluzione al problema è data da:

$$t_x = \phi_2(w_1, w_1', w_2, w_2') / \phi_d(w_1, w_1', w_2, w_2'), \quad (3)$$

$$t_y = -\phi_1(w_1, w_1', w_2, w_2') / \phi_d(w_1, w_1', w_2, w_2'), \quad (4)$$

$$r_{12} = \phi_3(w_1, w_1', w_2, w_2') / \phi_d(w_1, w_1', w_2, w_2'), \quad (5)$$

$$r_{11} = \phi_4(w_1, w_1', w_2, w_2') / \phi_d(w_1, w_1', w_2, w_2'). \quad (6)$$

# Il metodo di Groebner: Il caso spaziale

- Sebbene si è tentato di iterare lo stesso ragionamento, a causa di un elevato costo computazionale, non si è riuscito a completare il calcolo.

## Problema

data la soluzione del problema planare, è possibile estenderla al caso spaziale?

# Il metodo di Groebner: Il caso spaziale

## Punto centroide

Si consideri un set di punti  $\{w_1, w_2, \dots, w_N\}$ , si definisce **punto centroide** il punto centrale del set, calcolato come media puntuale dei punti.

$$w_c = \frac{1}{N} \sum_{i=1}^N w_i \quad (7)$$

## Asse del Mozzi

Sia considerata una trasformazione (rotazione e traslazione) di un corpo, si definisce **asse del Mozzi**, l'asse i cui punti, durante la trasformazione, subiscono una sola traslazione lungo lo stesso.

## Proprietà

- In caso di una sola rotazione lungo un asse passante per l'origine l'asse del Mozzi coincide con l'asse di rotazione stesso.
- L'asse del Mozzi può essere calcolato come l'intersezione di due piani omologhi, cioè dei piani definiti dagli stessi tre punti prima e dopo la rotazione rispettivamente.

## Il metodo di Groebner: Il caso spaziale

- Si costruiscano allora due set di punti  $W = \{w_1, w_2, w_3\}$  e  $W' = \{w'_1, w'_2, w'_3\}$
- Per ciascuno dei essi si calcoli il suo punto centroide  $w_c$  e  $w'_c$  rispettivamente.
- Si traslino i punti del set per far coincidere il rispettivo punto centroide con l'origine:

$$p_i = w_i - w_c, \quad p'_i = w'_i - w'_c \quad (8)$$

- Siano allora  $P = \{p_1, p_2, p_3\}$  e  $P' = \{p'_1, p'_2, p'_3\}$  i set da considerare.

# Il metodo di Groebner: Il caso spaziale

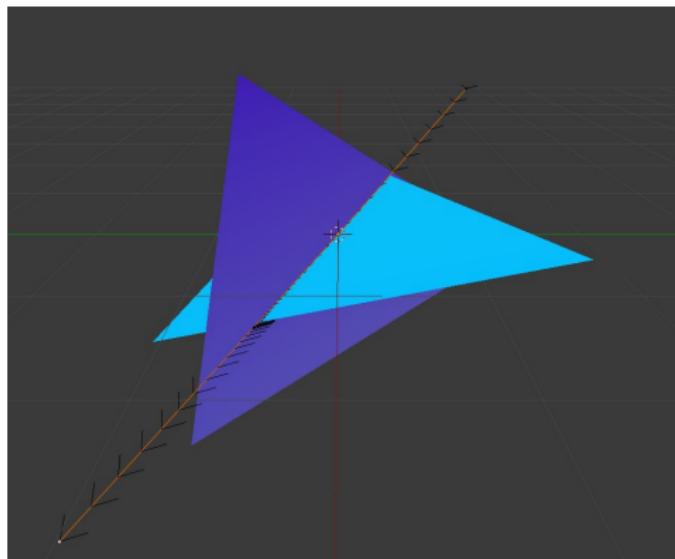


Figura: Calcolo dell'asse del Mozzi attraverso intersezione dei piani omologhi

## Il metodo di Groebner: Il caso spaziale

- Siano  $\hat{j}_1, \hat{j}_1' \in \mathbb{R}^2$  i versori della proiezione di una qualsiasi coppia omologa sul piano passante per l'origine e ortogonale all'asse del Mozzi. Si può calcolare l'angolo compreso tra i due versori:

$$\theta_1 = \arccos\left(\frac{\hat{j}_1 \cdot \hat{j}_1'}{\|\hat{j}_1\| \|\hat{j}_1'\|}\right) \quad (9)$$

- Si può definire la prima rotazione, utilizzando la rappresentazione asse-angolo come:

$$R_1 = \text{Rot}(\text{asse}_{\text{mozzi}}, \theta_1) \quad (10)$$

# Il metodo di Groebner: Il caso spaziale

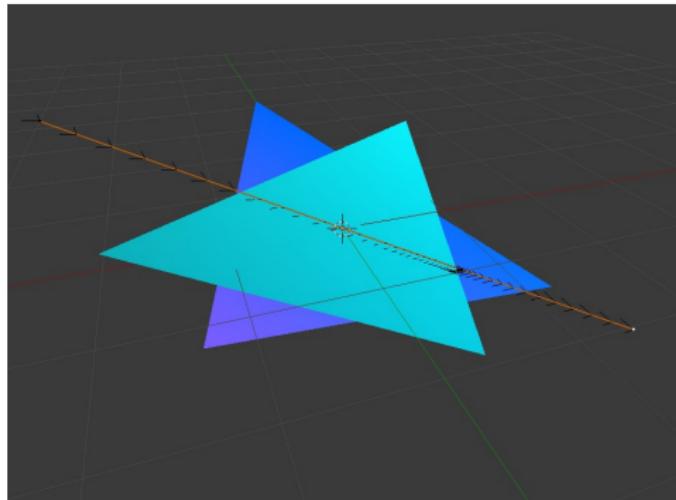


Figura: Dopo la rotazione sull'asse del Mozzi i due piani sono equi orientati

## Il metodo di Groebner: Il caso spaziale

- Si può utilizzare quanto fatto nel planare:

$$\cos(\theta_2) = \phi_4(R_1 p_1, p_1', R_1 p_2, p_2') / \phi_d(R_1 p_1, p_1', R_1 p_2, p_2'),$$
$$\sin(\theta_2) = \phi_3(R_1 p_1, p_1', R_1 p_2, p_2) / \phi_d(R_1 p_1, p_1', R_1 p_2, p_2'),$$

Si calcola l'angolo

$$\theta_2 = \text{atan}_2(\cos(\theta_2), \sin(\theta_2)) \quad (11)$$

- Si può definire la seconda rotazione, utilizzando la rappresentazione asse-angolo, come la rotazione intorno ad un asse unitario passante per l'origine e ortogonale ai piani.

$$R_2 = \text{Rot}(\text{asse}_{\perp \text{Piano}}, \theta_2) \quad (12)$$

## Il metodo di Groebner: Il caso spaziale

- Si ottiene la matrice di rotazione risultante pre-moltiplicando  $R_1$  per  $R_2$ , perché trasformazioni applicate in sequenza nel riferimento mobile solidale al primo piano.

$$R = R_2 R_1, R \in \mathbb{R}^{3 \times 3} \quad (13)$$

- Il vettore di traslazione può essere calcolato confrontando i centroidi al netto della rotazione:

$$t = w_c' - R w_c, t \in \mathbb{R}^d \quad (14)$$

- la matrice di trasformazione può essere calcolata come una roto-traslazione.

## Kabsch's Algoritmh

- presentato per la prima volta nel 1974 ad opera di W. Kabsch
- algoritmo di tipo: adattativo
- utilizzo tipico: bio-informatica confronto tra proteine

# Il metodo di Kabsch

- Siano dati due set di punti omologhi  $a_1, \dots, a_N$  e  $b_1, \dots, b_N$
- Si procede traslando i set di punti facendo si che i centroidi coincidano con l'origine:

$$x_i = a_i - a_c, \quad y_i = b_i - b_c \quad (15)$$

- Si fissa quindi l'indice di costo  $J = d(x, y)$

$$d(x, y) = \frac{1}{N} \sum_{i=1}^N \|Rx_i - y_i\|^2 \quad (16)$$

# Il metodo di Kabsch

- Si possono rappresentare i set di punti attraverso le matrici:

$$X = [x_1 \ x_2 \ \dots \ x_N], \ Y = [y_1 \ y_2 \ \dots \ y_N], \ X, \ Y \in \mathbb{R}^{d \times N}, \quad (17)$$

e sia  $X' = RX$ ,  $X' = [x'_1 \ x'_2 \ \dots \ x'_N]$  la matrice dei punti ruotati.

- Possiamo allora scrivere che:

$$NJ = \sum_{i=1}^N \|x'_i - y_i\|^2 = \text{Tr}[(X' - Y)^T(X' - Y)] \quad (18)$$

# Il metodo di Kabsch

- Grazie alla linearità dell'operatore traccia vale che:

$$Tr[(X' - Y)^T (X' - Y)] = Tr(X'^T X') + Tr(Y^T Y) - 2 Tr(Y^T X') \quad (19)$$

- Inoltre vale l'identità:

$$Tr(X'^T X') + Tr(Y^T Y) = \frac{1}{N} \sum_{i=1}^N \|x'_i\|^2 + \|y_i\|^2 \quad (20)$$

che grazie alla traslazione dei centroidi sull'origine è costante e può essere eliminata dall'indice di costo.

# Il metodo di Kabsch

- Per risolvere il problema allora basterà massimizzare la quantità

$$Tr(Y^T X') = Tr(Y^T RX) = Tr((XY^T)R) \quad (21)$$

- La matrice  $C = XY^T$ ,  $C \in \mathbb{R}^{d \times d}$  è la matrice di cross correlazione tra i due set.
- Si usa ora la decomposizione SVD della matrice  $C = VSW^T$ , si ha che  $V, S, W^T \in \mathbb{R}^{d \times d}$  poiché  $C$  è quadrata. La matrice  $S$  è diagonale e i suoi elementi sono i valori singolari della matrice  $C$  in ordine decrescente.

## Il metodo di Kabsch

- Dato che per l'operatore traccia vale che  $Tr(AB) = Tr(BA)$ , indicando con  $A = V$  e  $B = SW^T R$  è possibile scrivere:

$$Tr(Y^T X') = Tr(VSW^T R) = Tr(SW^T RV) = \sum_{i=1}^d s_i w_i^T R v_i \quad (22)$$

- La matrice  $T = W^T RV$  è ortonormale, questo vuol dire che i suoi elementi non potranno avere modulo maggiore di 1. Pertanto indicando con  $T_{ii} = w_i^T R v_i$  gli elementi sulla diagonale di  $T$ , vale che

$$Tr(Y^T X') = \sum_{i=1}^d s_i T_{ii} \leq \sum_{i=1}^d s_i \quad (23)$$

pertanto al massimo il valore della quantità  $Tr(Y^T X')$  è pari alla somma dei valori singolari  $s_i$ .

# Il metodo di Kabsch

- Si deve fare in modo quindi che la matrice  $T$  sia una matrice identità. Grazie alla proprietà di  $V$  e  $W^T$  di essere ortonormali,  $VV^T = I$ ,  $W^TW = I$ , basterà scegliere:

$$R = WV^T \tag{24}$$

- Resta da risolvere un problema, cioè che in generale  $\det(R) = \pm 1$ , il caso negativo va evitato. Per fare ciò possiamo aggiungere il vincolo che  $\det(R) = 1$ .

# Il metodo di Kabsch

- Notando che  $s_1 \geq s_2 \geq \dots \geq s_d$  e che

$$Tr(Y^T X') = \sum_{i=1}^d s_i T_{ii} \quad (25)$$

possiamo individuare il massimo per questa quantità, rispettando il vincolo, con la scelta  $T_{11} = T_{22} = \dots = T_{d-1d-1} = 1, T_{dd} = -1$ .

- Infatti, grazie alla richiesta di ortonormalità, deve valere che i  $T_{ii} = \pm 1$ .

# Il metodo di Kabsch

- La soluzione si ottiene pertanto come:

$$R = W \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & & & \\ 0 & \cdots & 1 & 0 \\ 0 & \cdots & 0 & d \end{bmatrix} V^T, \quad d = \text{sign}(\det(WV^T)) \quad (26)$$

- Analogamente al caso precedente,

$$t = b_c - Ra_c, \quad t \in \mathbb{R}^d \quad (27)$$

dove  $b_c$  e  $a_c$  sono i centroidi precedentemente calcolati.

## Il metodo di Kabsch: robustezza ai disturbi

- Considerando  $\tilde{x}_i$  la i-esima misura disturbata, si ha che

$$x_c = \frac{1}{N} \sum_{i=1}^N \tilde{x}_i = \frac{1}{N} \sum_{i=1}^N x_i + e_i \longrightarrow \hat{x} + E[e] = \hat{x} \quad (28)$$

$$c_{ij} = \sum_{k=1}^N \tilde{X}_{ik} \tilde{Y}_{kj} \longrightarrow \sum_{k=1}^N X_{ik} Y_{kj} + (\dots) \begin{bmatrix} E[e_x] \\ E[e_y] \end{bmatrix} = \hat{c}_{ij} \quad (29)$$

dove con  $E[e]$  si indica il valore atteso del disturbo, supposto rumore bianco.

- Al crescere del numero di campioni i centroidi e la matrice di cross-correlazione calcolati tendono a quelli "veri". Il risultato stimato dall'algoritmo tende a quello vero.

## Problema

è possibile calcolare/misurare le coordinate, nel sistema di riferimento solidale all'osservatore mobile, di un insieme di punti omologhi appartenenti al mondo circostante? Il mondo circostante si suppone ignoto.

## Soluzione:

- Feature detection: FAST + BRIEF + BF matcher
- Stereovisione

# Features detection: che cos'è una feature?

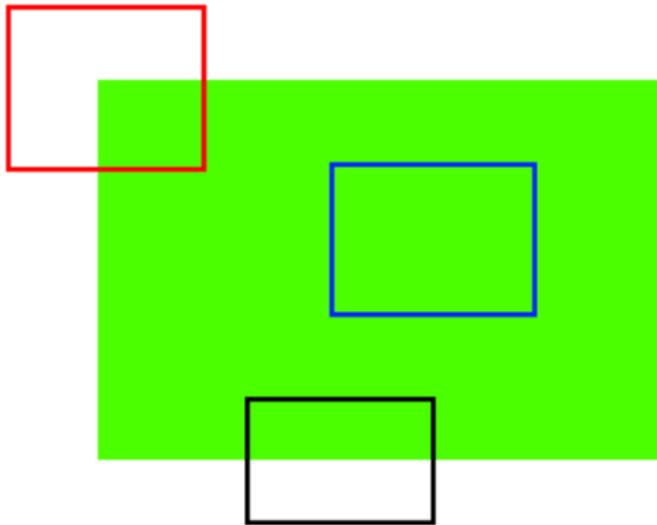


Figura: Individuazione delle Features.

## Features detection: FAST

- L'algoritmo FAST (Feature from Accelerated Segment Test) è adibito all'estrazione dei punti chiave.
- Caratteristiche: molto efficace, veloce e parallelizzabile.
- Funzionamento: ricerca spigoli nell'immagine.

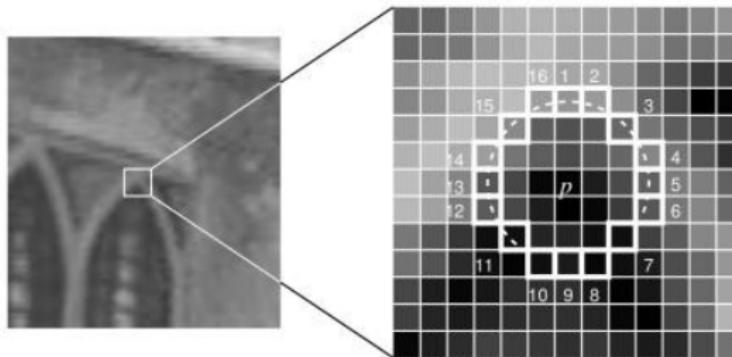
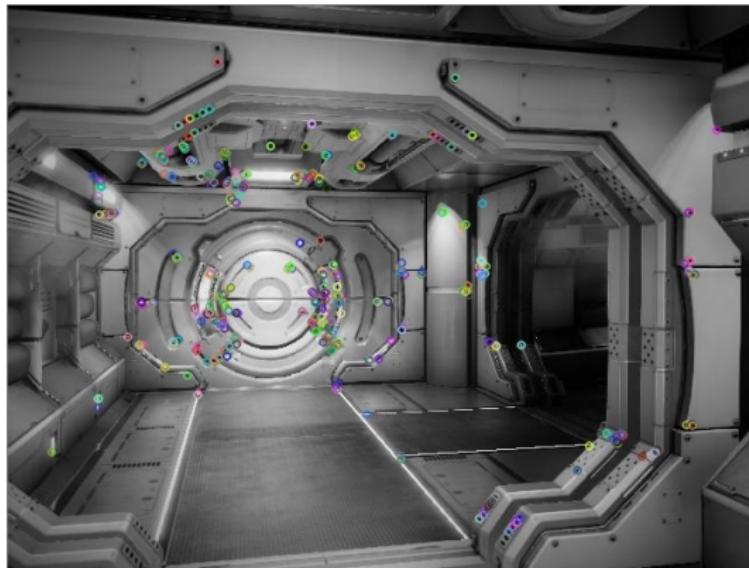


Figura: Fast Segment test.

# Features detection: FAST



**Figura:** Risultato dell'esecuzione dell'algoritmo FAST.

## Features description: BRIEF

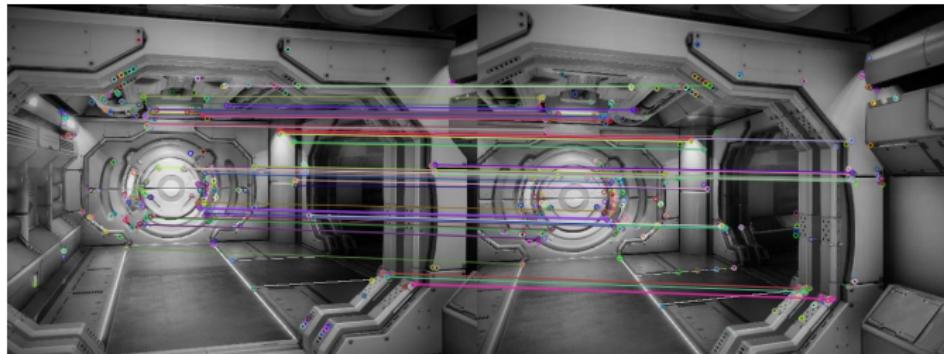
- L'algoritmo BRIEF, Binary Robust Independent Elementary Features, genera dei descrittori sotto forma di stringhe di bit.
- Il confronto tra due keypoint può essere fatto in modo estremamente efficiente, con una sola operazione XOR.
- pattern di confronto: uniforme, normale, randomico.
- Per ogni bit della stringa ed in base al pattern scelto se ne fissa il valore secondo la formula:

$$\tau_i(p, q_i) := \begin{cases} 1, & \text{se } I_p \leq I_{q_i} \\ 0, & \text{altrimenti} \end{cases} \quad (30)$$

dove con  $p$  si è indicato il pixel punto chiave,  $q_i$  l'i-esimo pixel scelto in base al pattern.

## Features matching: Forza bruta

- I confronti vengono effettuati su base dei descrittori con algoritmo a forza bruta.
- Vantaggio: facilmente parallelizzabile.

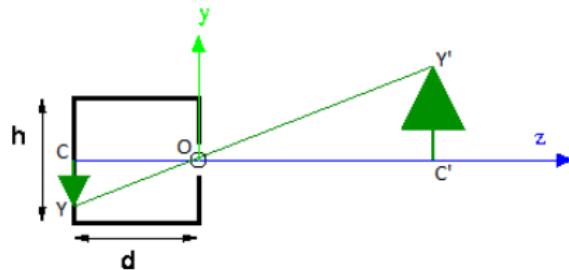


**Figura:** Risultato dell'esecuzione del comparatore: FAST + BRIEF + Brute Force matcher su due immagini in due punti di osservazione diversi

# Il sensore fotografico

Un sensore fotografico può essere modellato come nella seguente figura. Tale modello è detto **Pinhole camera**.

- Un singolo raggio di luce, da ogni angolazione, supera il pinhole e si infrange sul piano di cattura o film.



# Il sensore fotografico

## Modello **Pinhole camera sintetico**.

- Il piano di cattura viene capovolto.

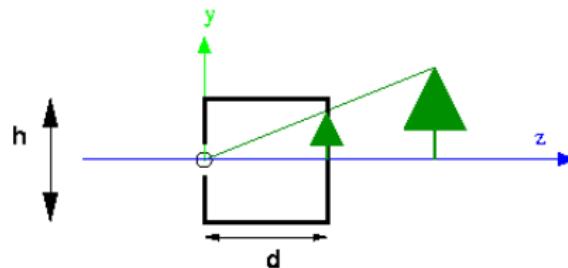


Figura: Esempio cattura immagine con sensore sintetico.

# Stereovisione

- Almeno due sensori sono necessari per la ricostruzione dell'informazione tridimensionale.

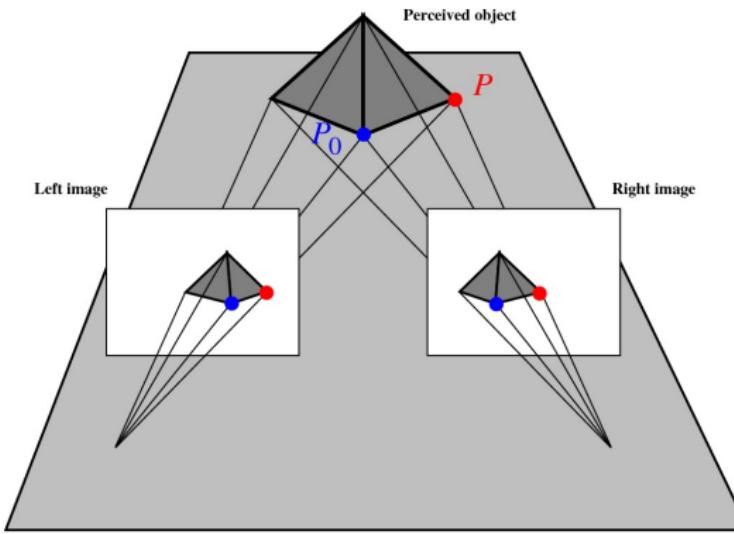


Figura: Coppia di immagini catturate da due sensori differenti nella scena.

# Stereovisione: centro di fissazione all'infinito

- Il calcolo può essere effettuato ragionando per triangoli simili.

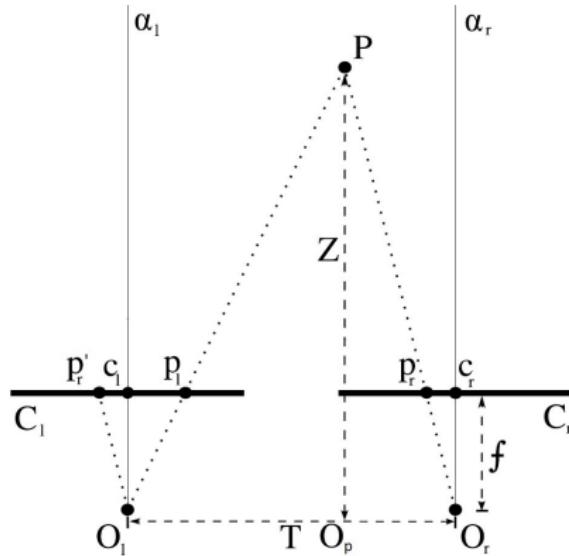


Figura: Configurazione sistema stereoscopico.

## Stereovisione: soluzione

- Si estraе per ciascun punto dalla coppia di immagini l'informazione di disparità, cioè la differenza in pixel della rispettiva coordinata X.

$$d := p_{I_x} - p_{r_x}, d \geq 0, d \in \mathbb{Z} \quad (31)$$

- Le coordinate di ciascun punto si calcolano nella seguente maniera:

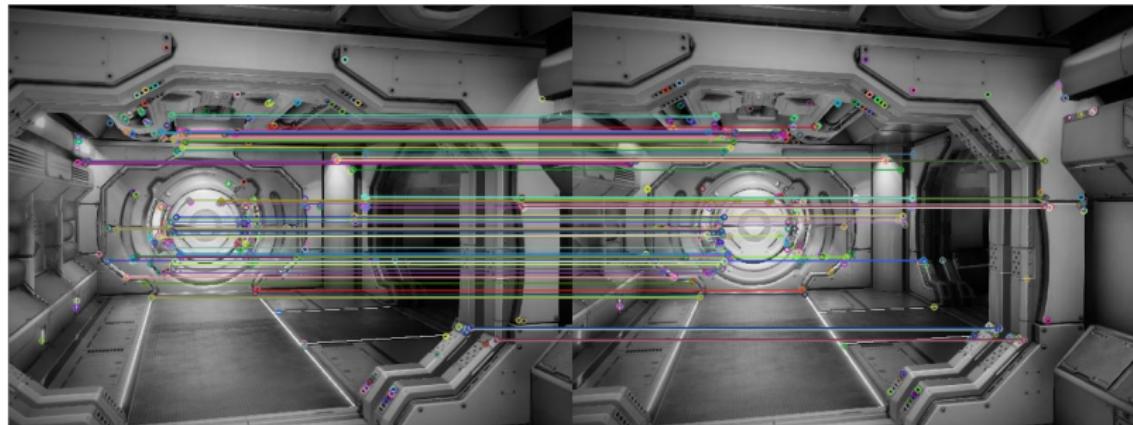
$$X := \frac{(p_{I_x} - c_{I_x})T}{d}, \quad (32)$$

$$Y := \frac{(p_{I_y} - c_{I_y})T}{d}, \quad (33)$$

$$Z := \frac{Tf}{d} \quad (34)$$

## Stereovisione: calcolo disparità

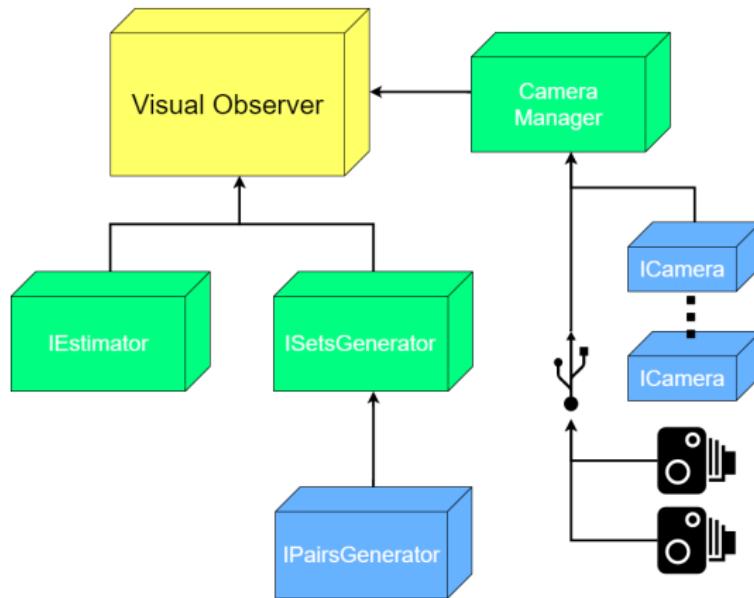
- Si può utilizzare ancora il features detection per individuare lo stesso punto nella coppia di foto stereo e calcolarne la disparità.



**Figura:** Risultato dell'esecuzione del comparatore: FAST + BRIEF + Brute Force matcher su una coppia di immagini stereo.

- **Piattaforma:** fissa e mobile,
- **Architettura target:** x86\_64, ARM,
- **Sistema Operativo target:** Windows, GNU/Linux,
- **Specifiche hardware minime:**
  - CPU: x86\_64 o ARM,
  - GPU: nessuna o generica con supporto OpenCL,
- **Specifiche hardware consigliate:**
  - CPU: x86\_64 o ARM con supporto OpenCL o SIMD/AVX,
  - GPU: coprocessore grafico Nvidia ad alte prestazioni con supporto CUDA Shared Model  $\geq 2.1$ ,
- **Linguaggio di sviluppo:** C++, OpenCV
- **Implementazioni:** CPU/SIMD, OpenCL, CUDA

# L'architettura



**Figura:** Rappresentazione architettura software Visual Observer.

# Ambiente virtuale



**Figura:** Ambiente virtuale generato con motore grafico Unity3D.

# Test effettuati

I test sono stati effettuati sulla seguente macchina:

- **CPU:** Intel i7-6700HQ @ 2.60 GHz,
  - **Architettura:** x86\_64,
  - **Specifiche:** AVX, AVX2, OpenCL,
- **Memoria:** 16 GB,
- **GPU:** Nvidia Geforce GTX 1070,
  - **Memoria dedicata:** 8 GB,
  - **Specifiche:** OpenCL, CUDA SM 6.1
- **Sistema Operativo:** Windows.

# Risultati

# Conclusioni

# Sviluppi futuri

Grazie

Grazie per l'attenzione!