UNIVERSITÀ DEGLI STUDI DI PADOVA

FACOLTÀ DI INGEGNERIA

CORSO DI LAUREA TRIENNALE IN INGEGNERIA MECCATRONICA

ALGORITMI DI OBJECT RECOGNITION PER PICCOLI ROBOT UMANOIDI

LAUREANDO: MACULAN MARCO

RELATORE: PROF. PRETTO ALBERTO

ANNO ACCADEMICO 2010/2011

Indice

1 Introduzione **5**

2 L'elaborazione delle immagini: segmentazione e rilevamento dei punti salienti **7**

2.1 Rilevamento e descrizione dei punti salienti: l'algoritmo SURF 7

2.2 Segmentazione delle immagini: l'algoritmo GrabCut 9

3 L'algoritmo proposto **10**

3.1 Panoramica dell'algoritmo 10

3.2 Fase di memorizzazione 10

3.3 Fase di apprendimento 10

3.3.1 Analisi dei punti notevoli ed estrazione dei descrittori 10

3.3.2 Clustering dei descrittori 11

3.3.3 Calcolo dei pesi di voto 11

3.4 Fase di ricerca 13

3.4.1 Estrazione dei descrittori e matching 13

3.4.2 Fase di voto 13

4 Implementazione **14**

4.1 Il Robot, simulazione tramite pc portatile 14

4.2 Integrazione con il Robot Operative System 15

4.2.1 Cos'è il ROS 15

4.2.2 Struttura del programma 15

4.3 Gli strumenti utilizzati 16

5 Esperimenti **17**

5.1 Acquisizione del dataset 17

5.2 Risultati 18

5.2.1 Descrizione del output 18

5.2.2 Descrizione del test principale di riconoscimento singolo 20

5.2.3 Risultati del test principale 21

5.2.4 Test secondario di riconoscimento multiplo 27

5.2.5 Test secondario di riconoscimento con occlusione parziale del soggetto 29

5.2.6 Risultati dei test secondari 29

6 Conclusioni **30**

6.1 Giudizio sulle funzionalità 30

6.2 Sviluppi futuri 30

Sommario

L'object recognition è un ramo dell'informatica che si occupa del rilevamento e della classificazione degli oggetti tramite analisi di immagini o video. E' molto importante nella progettazione dei robot perché permette a queste macchine di interpretare il segnale video esterno e di comprendere l'ambiente che le circonda. Si tratta di un passo fondamentale per rendere il robot autonomo e capace di portare avanti compiti complessi senza l'intervento umano.

Lo scopo di questa tesi è l'implementazione di un algoritmo di object recognition che permetta a dei piccoli robot umanoidi di apprendere una serie di oggetti, e successivamente di poterli riconoscere anche se visti in contesti diversi ed in presenza di altri oggetti.

La principale difficoltà risiede nelle limitate capacità computazionali che questi robot posseggono, le quali, unite alla necessità di eseguire sul robot tutte le fasi della procedura di riconoscimento, richiedono necessariamente la creazione di un algoritmo veloce e efficiente. Ponendo come obiettivo principale la velocità di esecuzione del programma, sono state fatte alcune scelte importanti: si sono innanzitutto scelte, come sistema di rilevamento dei punti notevoli, le SURF [2], le quali sono state progettate appositamente per essere veloci. Un programma di segmentazione dell'immagine è stato impiegato per ridurre al minimo la presenza di descrittori non legati agli oggetti da memorizzare. Si è poi implementato un algoritmo di clustering per comprimere il database di questi descrittori, mantenendo al contempo le informazioni grazie ad un sistema di votazioni euristico.

Questo algoritmo contribuisce alla ricerca implementando sistemi efficienti di rilevazione dei punti con uno schema di voto euristico, il tutto sotto ambiente ROS.

1 Introduzione

L'idea di costruire robot sempre più autonomi ed intelligenti ha dato vita a molti rami di ricerca, uno dei quali è la **computer vision** applicata alla robotica, un ambito a cui si possono ricollegare tutti quei processi che elaborano in modo automatico l'immagine al fine di ricavarne informazioni. Da un'immagine, che per la macchina non è altro che un insieme di numeri, si riesce a capire che cosa si sta guardando, come sono disposti gli oggetti e come si sviluppa lo spazio di fronte al robot.

Qui entra in gioco l'**object recognition**, una categoria di algoritmi che si pongono come scopo quello di riconoscere all'interno della scena che si sta visualizzando la presenza o meno di oggetti. Ora, questa abilità in noi esseri umani è del tutto innata, ma per le macchine è ancora un compito molto difficile: per poter apprendere un oggetto, o una categoria di oggetti, al robot servono centinaia o addirittura migliaia di immagini in cui l'oggetto sia ritratto. Inoltre, quando si tratta di riconoscerlo in una scena, una parziale occlusione dell'oggetto, o il semplice fatto che se ne trovino due o più affiancati, può mettere in seria difficoltà la macchina. Questo ci fa comprendere quanto la strategia di riconoscimento del robot sia distante dai sistemi che utilizza l'uomo per comprendere ciò che gli sta attorno.

La ricerca è tutt'ora in corso, ed in questi ultimi anni molte innovazioni sono state implementate: Uno dei più popolari sistemi di object recognition è stato proposto nel 2001 da Viola e Jones *[3],* che applicarono un algoritmo di apprendimento Adaptive Boosting (AdaBoost) ad un set di punti estratti da delle immagini con il metodo di Haar: questo approccio si adatta molto bene alla ricognizione facciale, ed è tutt'ora largamente utilizzato.

Verso il 2003, nel campo della ricognizione e classificazione degli oggetti vengono introdotti con successo i metodi bag-of-words (BoW) (ad es: [4]): Sono metodi che hanno come scopo quello di rappresentare l'immagine con una serie di descrittori clusterizzati (es: SIFT) presi da un vocabolario generato offline.

Negli anni successivi questi metodi vengono ulteriormente migliorati indicizzando i descrittori tramite una struttura ad albero con una quantizzazione gerarchica [5] e utilizzando metodi di quantizzazione basati sulle strutture ad albero casuali [6] aggiungendo una fase di riassegnazione basata sulla verifica spaziale delle regioni di interesse.

Questi metodi ([5] e [6]) hanno il difetto di funzionare solo se si utilizzano vocabolari molto grandi, che possono richiedere molto sforzo computazionale. Pertanto nel 2005 Winn [8] ha introdotto il concetto di dizionario compresso, riducendo set di dati molto voluminosi a set più piccoli ma con un contenuto informativo molto simile.

Da questa idea si svilupparono nel 2006 i lavori di Marszalek e Schmid [7] e Leibe [9]. Quest'ultimo clusterizzò un gran numero di descrittori utilizzando un algoritmo combinato di partizione ed agglomerazione, riuscendo a velocizzare il processo di matching. Poi nel 2007 Lazebnik e Raginsky [10] partizionarono il loro vocabolario utilizzando come riferimento le celle di un diagramma di Voronoi.

In tutte queste applicazioni la creazione del dataset e l'esecuzione della fase di apprendimento sono compiute a parte, magari su macchine più potenti, trasferendo poi il risultato all'eventuale robot incaricato della fase di analisi.

Il lavoro qui proposto permette invece compiere queste operazioni online: Sia la creazione del dataset, che l'elaborazione dello stesso sono eseguite dal robot. Si è pertanto lavorato per rendere efficiente anche questa fase, in maniera che la sua esecuzione occupi un tempo ragionevole. Nella prossima parte verranno descritte le tecniche impiegate per l'elaborazione delle immagini e per l'estrazione dei loro punti salienti. Verrà poi illustrata la struttura dell'algoritmo, descrivendone le fasi ed i sistemi utilizzati. A seguire di descriveranno le caratteristiche di implementazione, spiegando dove si è pensato di implementare questo algoritmo, e che conseguenze ha avuto ciò sulle scelte di programmazione. Infine si parlerà di risultati, di come sono stati ottenuti e dei punti di forza e debolezze dell'algoritmo. A chiudere ci sarà la sezione delle conclusioni, dove si scriverà di possibili sviluppi futuri.

2 L'elaborazione delle immagini: segmentazione e rilevamento dei punti salienti

In questa sezione si spiega nel dettaglio il funzionamento degli algoritmi utilizzati per elaborare le immagini in ingresso, descrivendone pregi e difetti.

2.1 Rilevamento e descrizione dei punti salienti: l'algoritmo SURF

L'algoritmo SURF [2], acronimo di "Speeded Up Robust Features", nasce dalla necessità di rilevare e descrivere i punti salienti in tempi rapidi. Esso deriva dall'algoritmo SIFT [12], "Scale-Invariant Feature Transform" che ha la caratteristica di essere molto robusto, ma al contempo molto lento. Le SIFT infatti impiegano diversi secondi per analizzare una singola immagine, producendo per ogni punto un descrittore con 128 valori, cosa che rallenta enormemente anche la fase di matching.

Le SURF nascono con l'obiettivo di velocizzare la fase di rilevazione e descrizione, e contemporaneamente produrre un tipo di descrittori più piccolo, a scapito però di una minore robustezza in caso di forti cambi di illuminazione e di rotazione del soggetto.

Di seguito si possono notare i risultati di alcuni confronti eseguiti tra Surf e Sift:

	SURF	**SIFT**
Memoria occupata	SURF: 64 floats SURF-128: 128 floats	128 bytes
Velocità (Tempo di rilevamento e descrizione di 4000 punti)	2,4 secondi	6 secondi
Punti localizzati in un immagine di 1024 x 768	~ 1000	> 3000

Tabella 1: Caratteristiche di analisi a confronto

Il metodo adottato per aumentare significativamente la velocità di localizzazione si basa sull'analisi dell'immagine integrata, una rappresentazione dell'immagine originale che contiene la somma del valore di grigio dei pixel dell'immagine in una data regione rettangolare.

Dopodiché a questa approssimazione dell'immagine originale viene calcolato il determinante della matrice hessiana. Ciò fa risaltare molto i cambi di intensità luminosa, evidenziando gli angoli ma non i bordi degli oggetti. Per velocizzare il processo si usano dei kernel che eseguono una media pesata dell'intensità del pixel.

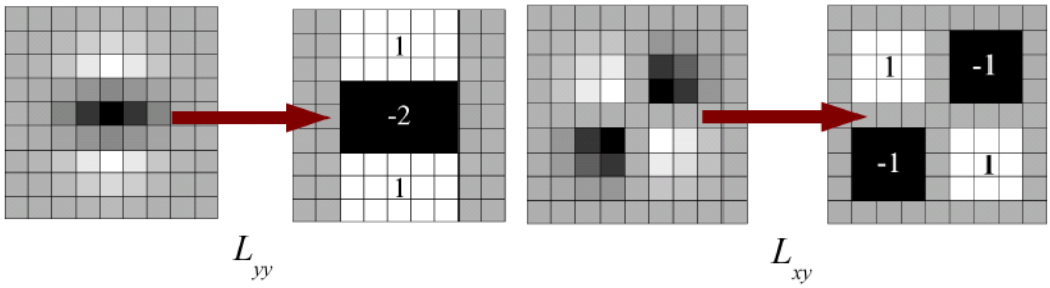


Figura 1: Approssimazione dei kernel

Qui sotto sono messe a confronto varie tecniche di rilevamento. Per il programma si è utilizzato il metodo Fast-Hessian.

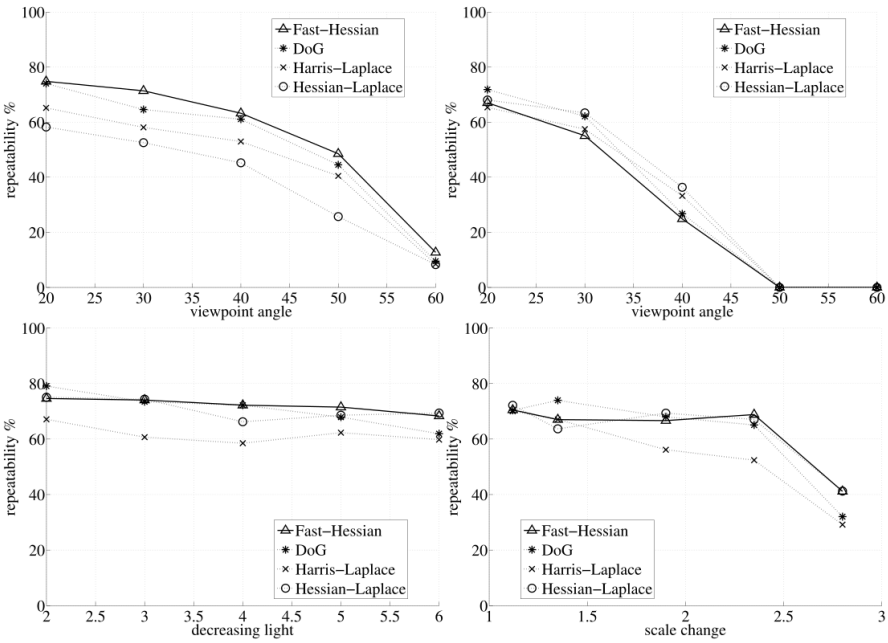
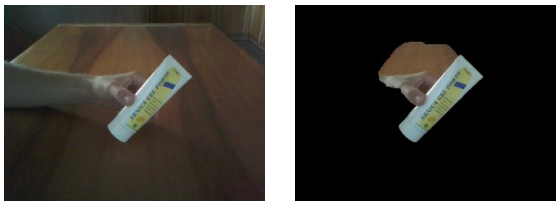


Figura 2: Metodi di rilevazione dei punti notevoli a confronto in caso di cambio di visuale, cambio di illuminazione e variazione di scala.

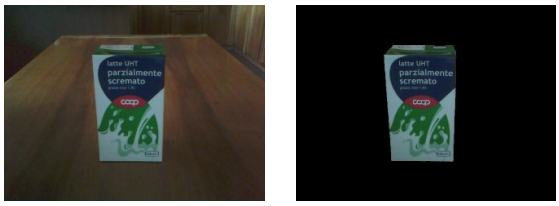
2.2 Segmentazione delle immagini: l'algoritmo GrabCut

GrabCut [1] è un algoritmo di segmentazione che ha come scopo quello di isolare il soggetto dell'immagine dallo sfondo. Per fare ciò il programma cataloga i pixel dell'immagine in quattro categorie: pixel di background certi e presunti, pixel del soggetto certi e presunti.Il funzionamento si basa su un processo di minimizzazione di una funzione energia che ha come parametri di ingresso la scala di grigio e la componente colorata del pixel. Il programma è pensato per essere iterato più volte, ma in fase di sviluppo si è notato che una iterazione è sufficiente per isolare il soggetto. L'importante è considerare come punti di fondo anche quelli che l'algoritmo classifica come possibili punti di background.

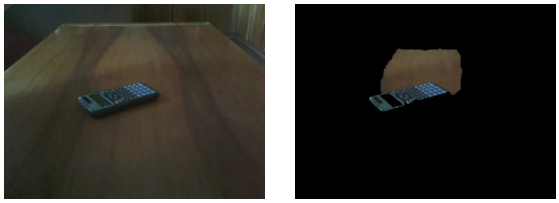
Nelle figure sottostanti è possibile notare che con un solo passaggio la maggior parte dello sfondo viene rimossa. Nei primi tre casi il risultato è sicuramente accettabile, mentre nel quarto caso la zona di interesse viene quasi del tutto perduta. La causa è imputabile principalmente alla scelta di uno sfondo di colore troppo simile al soggetto.



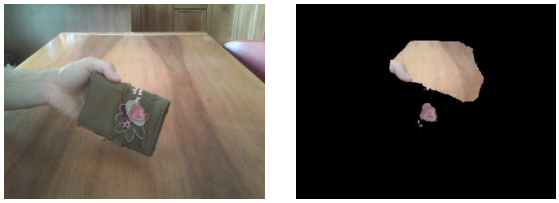
Taglio parziale



Taglio preciso



Taglio abbondante



Taglio errato

3 L'algoritmo proposto

3.1 Panoramica dell'algoritmo

L'algoritmo si articola in tre fasi: una fase di memorizzazione in cui si immagazzinano le immagini degli oggetti, la fase di apprendimento in cui le informazioni sugli oggetti vengono estratte ed elaborate, ed infine la fase di ricerca, in cui si analizza la scena e la si confronta con i dati immagazzinati. Le prime due fasi devono necessariamente essere eseguite in sequenza per creare un dataset utilizzabile. Se si è già in possesso del dataset allora è possibile passare direttamente alla fase di ricerca, che è totalmente svincolata dall'esecuzione delle prime due fasi.

3.2 Fase di memorizzazione

In questa fase le immagini vengono memorizzate tramite webcam e catalogate per oggetto. Si deve avere l'accortezza di riprendere l'oggetto in primo piano e da più angolature e punti di vista possibili, in modo da ottenere un buon dataset.

3.3 Fase di apprendimento

3.3.1 Analisi dei punti notevoli ed estrazione dei descrittori

Innanzitutto il programma controlla che non sia già presente la versione ritagliata dell'immagine. Se non esiste, le immagini originali vengono ritagliate in maniera da eliminare lo sfondo e mantenere solamente l'oggetto che si ha interesse ad analizzare. Per fare ciò si utilizza GrabCut, che genera una maschera da applicare all'immagine per ottenere solamente il soggetto in primo piano.

Poi l'algoritmo esamina le immagini processate identificandone i punti di interesse ed estraendo i descrittori tramite l'algoritmo SURF di corner detection.

3.3.2 Clustering dei descrittori

Dato che l'obiettivo principale è la velocità, un passaggio obbligato è quello del clustering, che è stato effettuato con K-means [13]. In questa fase tutti i descrittori trovati vengono divisi in gruppi denominati cluster ed assegnati ad un punto detto centro del cluster o centroide, usando come criterio la loro distanza euclidea relativa. Per fare ciò si tratta il descrittore, che non è altro che un vettore di *n* valori, come se fosse un punto *n*-dimensionale. Nel nostro caso si tratta di descrittori a 64 valori, pertanto andiamo a raggruppare nuvole di punti in uno spazio a 64 dimensioni. Dato che queste nuvole sono molto compatte, e la maggior parte di questo spazio è vuoto, con un buon clustering si riesce a ridurre di molto il numero di descrittori, approssimando ogni punto della nuvola al baricentro della stessa.

L'algoritmo segue una procedura iterativa finché nessun punto cambia cluster, oppure finché la differenza della distanza media dei punti dal loro centro assegnato tra due iterazioni non cala al di sotto di una data soglia. Inizialmente l'algoritmo crea K partizioni e assegna ad ogni partizione i punti in maniera casuale o usando alcune informazioni euristiche. Quindi calcola il centroide di ogni gruppo. Costruisce quindi una nuova partizione associando ogni punto al cluster il cui centroide è più vicino ad esso. Quindi vengono ricalcolati i centroidi per i nuovi cluster. Il tutto viene ripetuto finché l'algoritmo non converge. A questo punto però si rende necessario anche un metodo per definire che tipo di descrittori stia approssimando ogni centroide, e a che oggetti esso si stia riferendo. Qui entra in gioco il calcolo dei pesi di voto.

3.3.3 Calcolo dei pesi di voto

Il calcolo dei pesi di voto è un'euristica che si pone come obiettivo quello di calcolare un set di pesi proporzionale alla probabilità che dato un certo punto notevole ci sia in scena un determinato oggetto. Per fare ciò ci si è appoggiati alle leggi della teoria Bayesiana (vedi appendice A).

Più precisamente per ogni centroide viene calcolata una serie di pesi, uno per ogni oggetto. L' *i*-esimo numero di questo vettore rappresenta un peso proporzionale alla probabilità che, rilevato il centroide nella scena, l'oggetto *i*-esimo sia presente.

Per calcolare $p(O_i|F_k)$[1] si sono usate le proprietà della teoria Bayesiana:

$$p(O_i|F_k) = \frac{p(F_k|O_i) \cdot p(O_i)}{p(F_k)}$$

Le tre componenti dell'equazione hanno i seguenti significati:

- $p(F_k|O_i)$ Probabilità condizionata che appaia F_k nota la presenza in scena di O_i;
- $p(O_i)$ Probabilità a priori che appaia O_i;
- $p(F_k)$ Probabilità a priori che appaia F_k;

Le tre componenti dell'equazione si sono calcolate così:

$$p(F_k|O_i) = \frac{P_{ik}}{P_i}$$

dove P_{ik} è pari al numero di punti notevoli appartenenti all'oggetto *i* che si trovano nel cluster del centroide *k* e P_iè pari al totale dei punti notevoli rilevati per l'oggetto.

$$p(O_i) = \frac{1}{n}$$

Dove *n* è il numero di oggetti.

$$p(F_k) = \frac{P_k}{P}$$

dove P_k è pari al numero di punti notevoli che si trovano nel cluster del centroide *k* e P è pari al totale dei punti notevoli rilevati per tutti gli oggetti.

Infine si ottiene:

[1]Con $p(x)$ si intende un valore numerico da zero a uno che indica la probabilità che l'evento *x* accada a priori.

Con$p(x|y)$ si intende la probabilità condizionata che l'evento *x* accada, noto *y*.

$$p(O_i|F_k) = \frac{P_{ik} \cdot P}{P_i \cdot P_k \cdot n}$$

Il valore $p(O_i|F_k)$ diviene infine il voto che il punto F_k dà all'oggetto O_i.

3.4 Fase di ricerca

3.4.1 Estrazione dei descrittori e matching

In questa modalità il robot esamina le immagini in entrata e vi estrae i punti notevoli e relativi descrittori, con metodi analoghi a quelli descritti sopra. Poi si collega ogni descrittore con il centro di clustering più vicino, utilizzando sempre il criterio di distanza euclidea.

Vengono poi tenuti solamente le coppie con una distanza minore di due volte la distanza media euclidea dei descrittori dai relativi centri di cluster. In questo modo si eliminano dall'analisi tutti quei punti che ragionevolmente non appartengono a nessun oggetto.

3.4.2 Fase di voto

In questa fase si analizzano tutti i centroidi associati ai punti notevoli della scena. Per ogni oggetto tutti i centroidi votano con il valore precedentemente calcolato per quell'oggetto. Il voto finale è una **media quadratica** dei voti pesata secondo la quantità di punti di scena associati al singolo centroide. Se il voto finale di un oggetto supera una certa soglia *s*, definita sperimentalmente come $s = \frac{1{,}8}{n}$, allora il programma dichiara che quell'oggetto è presente in scena. La media quadratica si rende necessaria per ottenere un buon rilevamento: infatti spesso quando si analizza una scena con un oggetto che si è memorizzato, la maggior parte dei punti trovati non apparterrà a nessun oggetto in particolare, e darà dei voti bassi, anche all'oggetto in questione. sono una minoranza di punti darà voti alti all'oggetto, e in una media normale non farebbero la differenza. Utilizzando invece una media quadratica, i valori alti vengono amplificati, permettendo una ricognizione dell'oggetto più solida.

4 Implementazione

4.1 Il Robot, simulazione tramite pc portatile

Inizialmente il progetto prevedeva l'implementazione di tutto il sistema su un robot umanoide autonomo Robovie-X, che doveva essere pilotato tramite un Limestone PDA Kit, una scheda di controllo con processore da 806 Mhz.

La scheda è stata ordinata, ma purtroppo l'arrivo in laboratorio è previsto per ottobre 2011, per cui si è deciso di simulare il robot tramite un pc portatile dotato di webcam. Per fare ciò si è installato sul pc il Robot Operative System, che verrà illustrato più avanti.

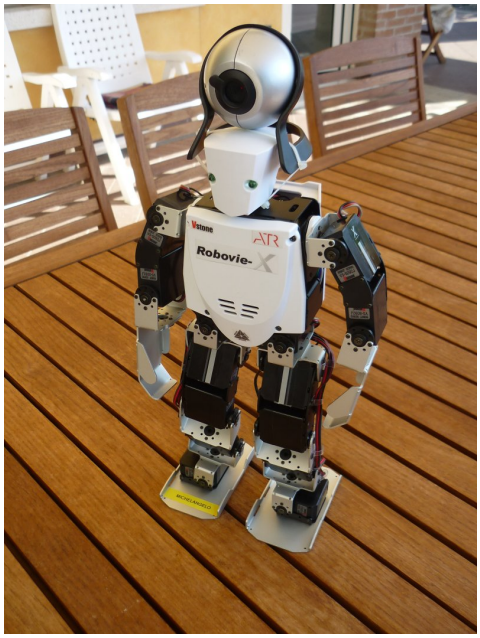


Figura 3: Il robot Robovie - X

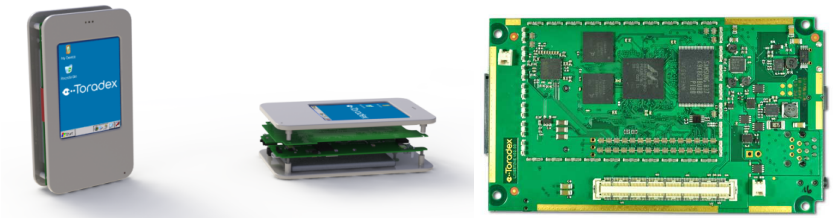


Figura 4: Limestone PDA Kit

4.2 Integrazione con il Robot Operative System

4.2.1 Cos'è il ROS

Il ROS [11], (Robot Operative System) è un framework Open Source che sta diventando lo standard *de facto* tra i sistemi di controllo per robot. Attualmente funziona su sistemi operativi Unix, e fornisce una serie di funzionalità tipiche di un sistema operativo. Caratteristica principale di questo sistema è quella di essere strutturato in nodi tra loro autonomi, che possono scambiarsi dati tramite canali appositi. Ogni nodo lo si può vedere come un processo a sé, pertanto la messa in atto di strategie multithread ne risulta agevolata.

Si è deciso di implementare l'algoritmo con questo framework, in maniera da renderlo facilmente esportabile anche su altri robot. Come distribuzione ROS si è scelta la Diamondback.

4.2.2 Struttura del programma

Un nodo, chiamato "uvc_cam", legge dalla webcam e trasmette sul canale "image_raw" il segnale in uscita. Il nodo "ObjRec" ha il compito di leggere le immagini dal canale, elaborarle, e trasmettere il risultato sul canale "ImmagineProc", che viene prontamente interpretato dal nodo "image_view", il quale ha come compito la visualizzazione a schermo delle immagini.

Ma il nodo "ObjRec" si fa anche carico di tutte le fasi dell'algoritmo, dalla memorizzazione all'analisi, e compie tutto ciò utilizzando un sistema a stati: Il nodo può trovarsi in modalità apprendimento, modalità elaborazione, o in modalità analisi. Ciò che pilota il cambio da uno stato ad un altro è un ulteriore nodo, denominato "nodoDiControllo", tramite un apposito canale "canaleControlloObjRec", sul quale "ObjRec" si trova sempre in ascolto.

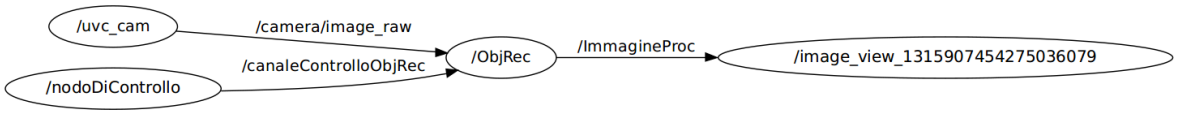


Figura 5: Grafo del programma

4.3 Gli strumenti utilizzati

Il cuore del programma è stato scritto in linguaggio C++, usando come sistema operativo la distribuzione linux Ubuntu e Kdevelop come IDE di programmazione.

Per la manipolazione delle immagini e l'estrazione dei punti notevoli con relativi descrittori si sono utilizzate principalmente le librerie OpenCV [14]. Queste librerie open source sono sviluppate appositamente per la computer vision, ed al loro interno si possono trovare molte funzioni e tipi di dati utili per manipolare immagini, ed inoltre hanno il vantaggio di essere già implementate in ROS. Difatti, esiste un pacchetto apposito che permette di utilizzare tutte le funzionalità di queste librerie all'interno dell'ambiente di lavoro ROS.

Per generare i grafici ci si è appoggiati invece alle librerie grafiche Gnuplot [15].

5 Esperimenti

5.1 Acquisizione del dataset

Per acquisire online il dataset il robot utilizza la webcam per catturare una serie di immagini che ritraggono l'oggetto in primo piano, su fondo possibilmente neutro. Le immagini vanno scattate da angolature e distanze differenti, in maniera da ritrarre l'oggetto in più pose possibili. Una volta che se ne sono sono catturate abbastanza, si può memorizzare un altro oggetto oppure passare alla fase di apprendimento.

Per creare il dataset su cui compiere i test sono stati scelti 8 oggetti di uso comune. Ruotandoli e cambiando angolazione si sono memorizzate 100 immagini per ogni oggetto. Al termine della procedura è stata lanciata la fase di elaborazione, che ha ritagliato tutte le immagini, ne ha esaminato i punti salienti e ha compiuto un clustering finale sulla totalità dei descrittori trovati.

Figura 6: Panoramica degli oggetti memorizzati

Oggetto	Numero descrittori
Shower gel	21457
Latte	28674
Calcolatrice	13468
Tea	21021
Tazza	28181
Libro	25734
Portafoglio	20488
Gomme	27692
Totale	**186715**

Tabella 2: Numero di descrittori rilevati

5.2 Risultati

5.2.1 Descrizione del output

Il programma restituisce in uscita l'immagine esaminata a cui vengono aggiunte informazioni aggiuntive sull'analisi fatta: in alto a sinistra viene stampata una frazione il cui numeratore indica il numero di punti notevoli che si trovano sotto la distanza minima dal centroide a loro più vicino. Il denominatore indica invece il totale dei punti rilevati sulla scena.

Vengono inoltre cerchiati i punti notevoli con un colore che vira dal blu al rosso a seconda di quanto essi siano vicini al centroide assegnato. Infine vengono stampati in basso i risultati delle votazioni.

Il programma produce inoltre un istogramma con i voti, in maniera da rendere immediatamente visibile il risultato della votazione, e se uno o più oggetti superano la soglia di rilevazione, il loro nome viene scritto poco sopra i voti.

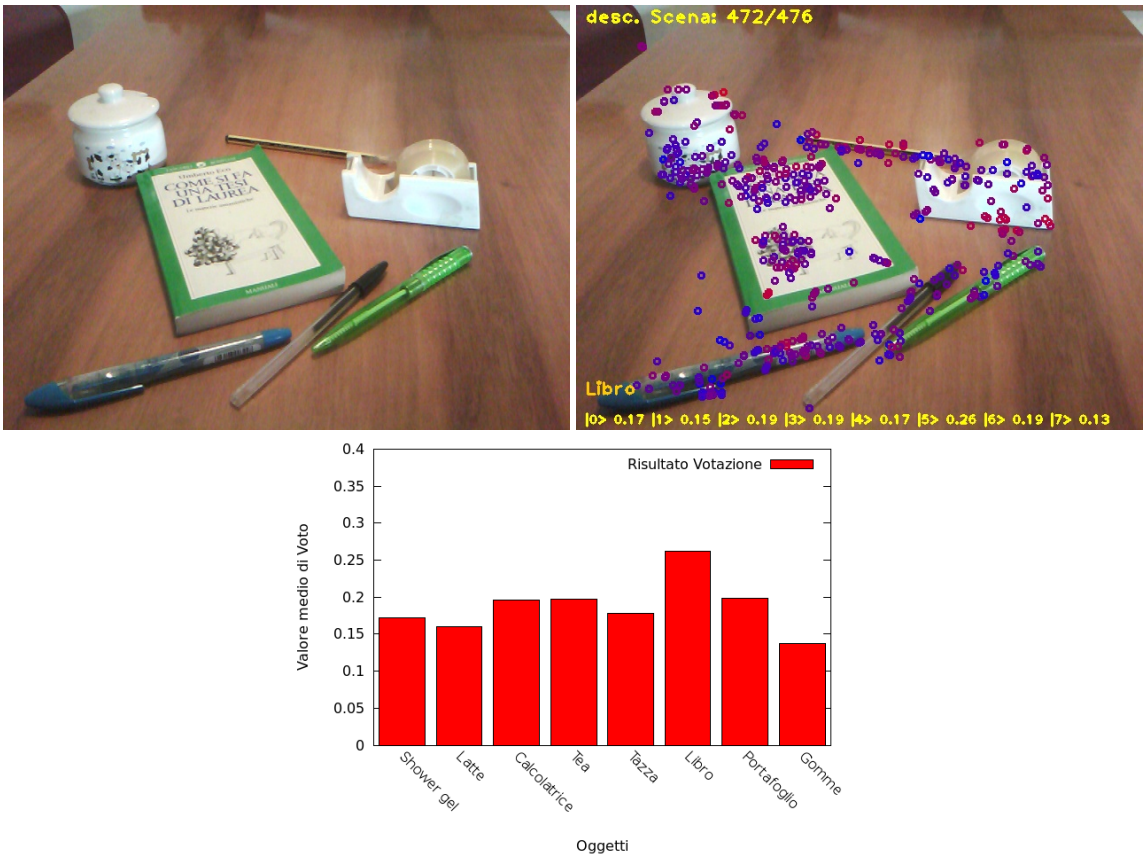


Figura 7: Esempio di risultato

5.2.2 Descrizione del test principale di riconoscimento singolo

Per testare il programma sono state scattate delle immagini di prova e sono state passate al nodo principale come se fossero dei frame in arrivo dalla webcam, salvando poi i risultati sul disco. Per simulare la ricerca da parte di un robot, si è composta una scena, e si è poi mosso il pc in modo da riprendere la scena da più punti di vista. Attraverso la webcam sono state catturate 5 immagini per oggetto e sono state poi processate dall'algoritmo.

Figura 8: Metodo di test

5.2.3 Risultati del test principale

La tabella sottostante raccoglie alcune delle 40 immagini elaborate, a seguire una tabella riassume i dati salienti del test.

Tabella 3: Immagini dei risultati

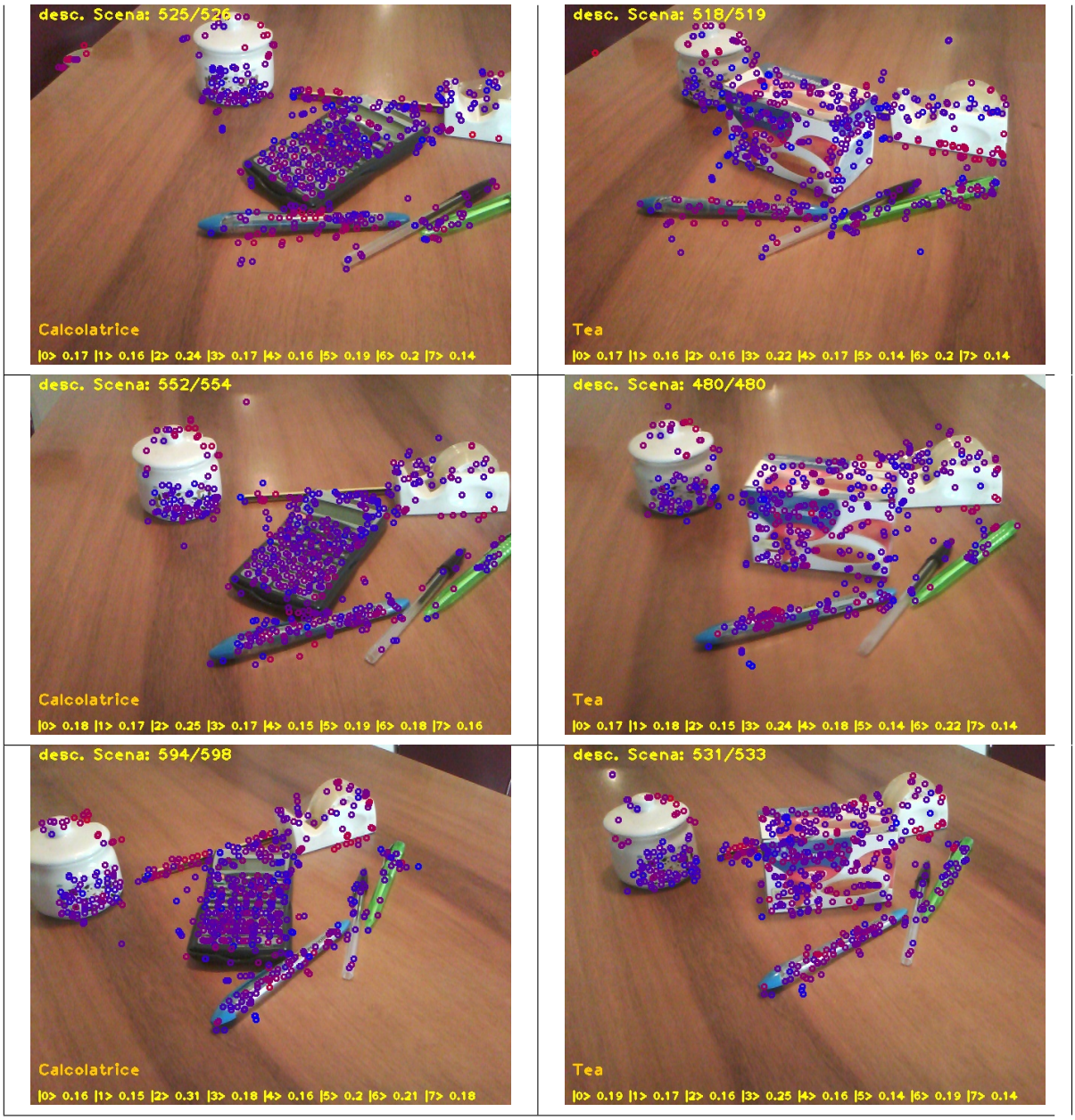


Tabella 4: Immagini dei risultati

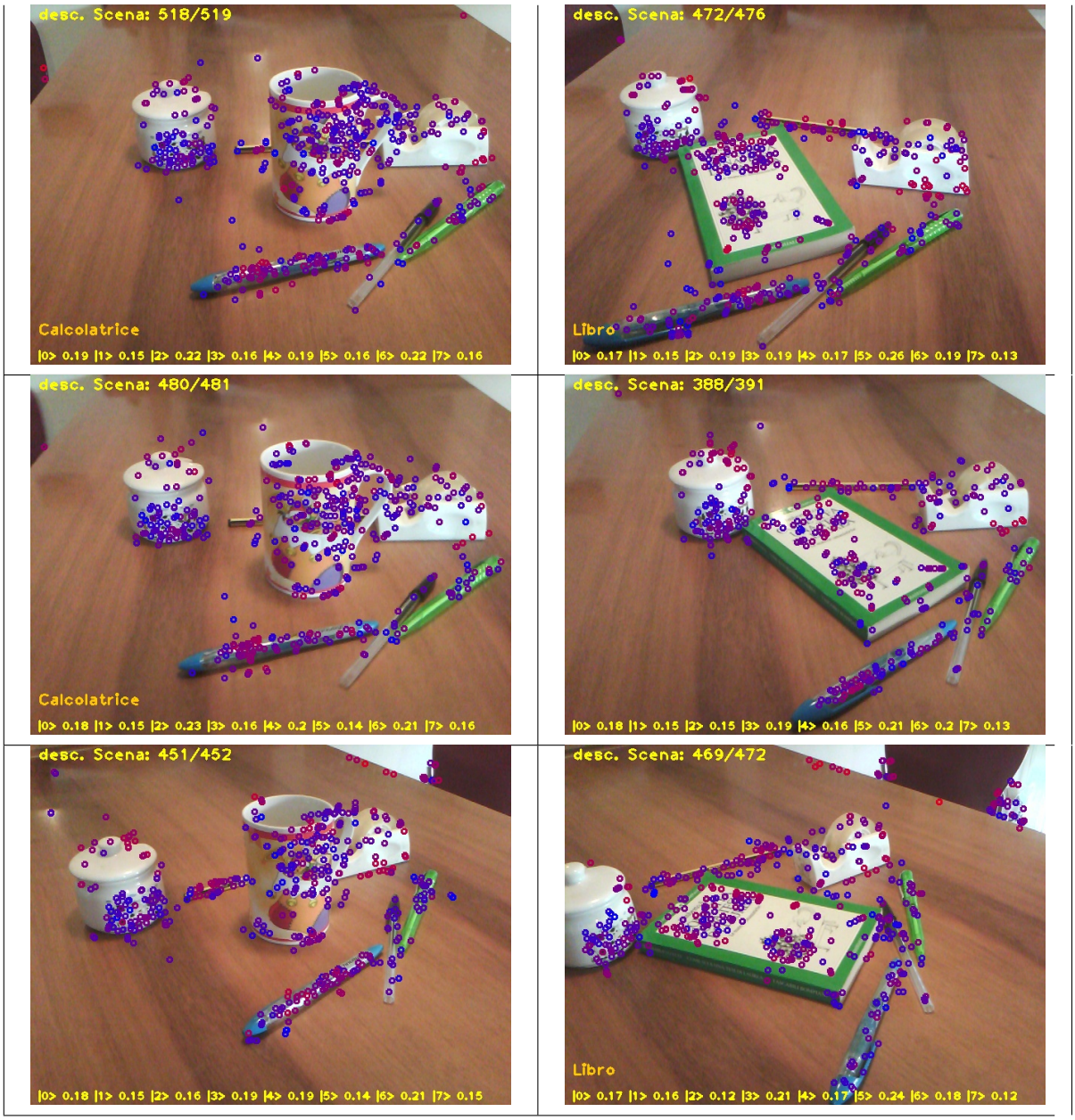


Tabella 5: Immagini dei risultati

Tabella 6: Immagini dei risultati

Oggetto	Rilevazioni positive	Falsi positivi
Shower gel	5 / 5	0
Latte	5 / 5	0
Calcolatrice	5 / 5	1
Tea	5 / 5	0
Tazza	0 / 5	2
Libro	3 / 5	0
Portafoglio	5 / 5	0
Gomme	3 / 5	0

Tabella 7: Riassunto dei risultati

Si può notare che nella maggioranza dei casi l'oggetto viene riconosciuto, e solo in un caso il test è fallito. Per scoprire la causa è stato necessario analizzare i valori di voto che ogni descrittore attribuisce ad un determinato oggetto. Si è ricavata la media e lo scarto quadratico medio dei valori dalla media per gli oggetti più interessanti. Ecco i risultati:

Oggetto	Media	Scarto quadratico medio
Calcolatrice	0.127680	0.207951
Libro	0.125958	0.192911
Portafoglio	0.118897	0.182894
Tazza	0.131364	0.133358

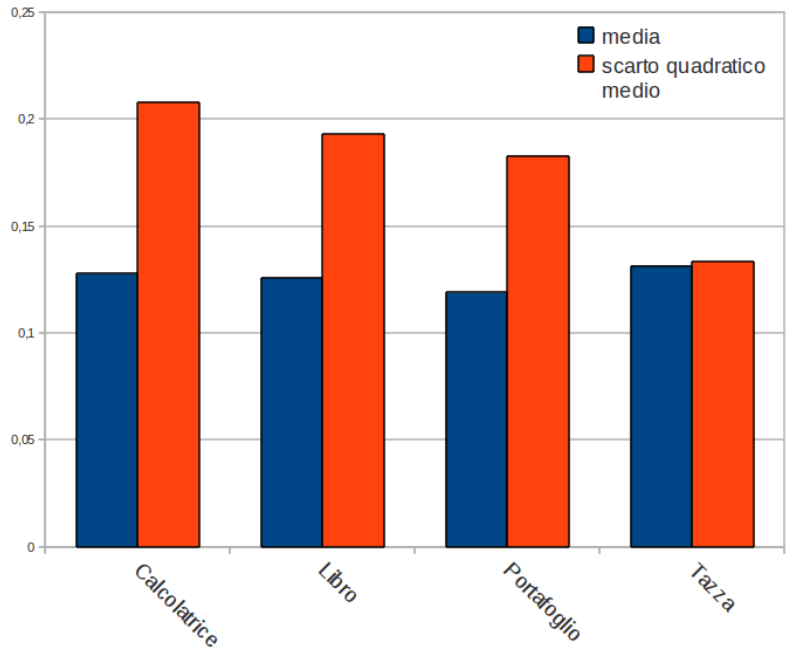


Tabella 8: Risultato delle analisi sui voti

Alcuni oggetti, come il portafoglio o la calcolatrice si dimostrano particolarmente robusti, per il fatto che posseggono punti notevoli molto caratteristici. Difatti il loro scarto quadratico è molto alto, segno che questi oggetti hanno punti salienti che pochi altri oggetti possiedono, e che conferiscono all'oggetto un voto alto.

La tazza invece possiede uno scarto quadratico più basso, segno che i descrittori danno tutti un voto simile alla tazza. Cio significa che non ci sono punti caratterizzanti che permettano all'oggetto di spiccare sugli altri, giustificando perciò il risultato del test.

5.2.4 Test secondario di riconoscimento multiplo

Sono state poi eseguite ulteriori acquisizioni per testare la capacità di riconoscimento multiplo dell'algoritmo, inserendo in scena due oggetti appartenenti al database. Il problema principale del riconoscimento multiplo è che gli oggetti si contendono la scena danneggiando l'uno il voto dell'altro. Infatti, un punto fortemente caratterizzante per un oggetto apporta di conseguenza un contributo molto piccolo alle votazioni di altri oggetti, a differenza di punti di sfondo neutri che votano in egual misura tutti gli oggetti.

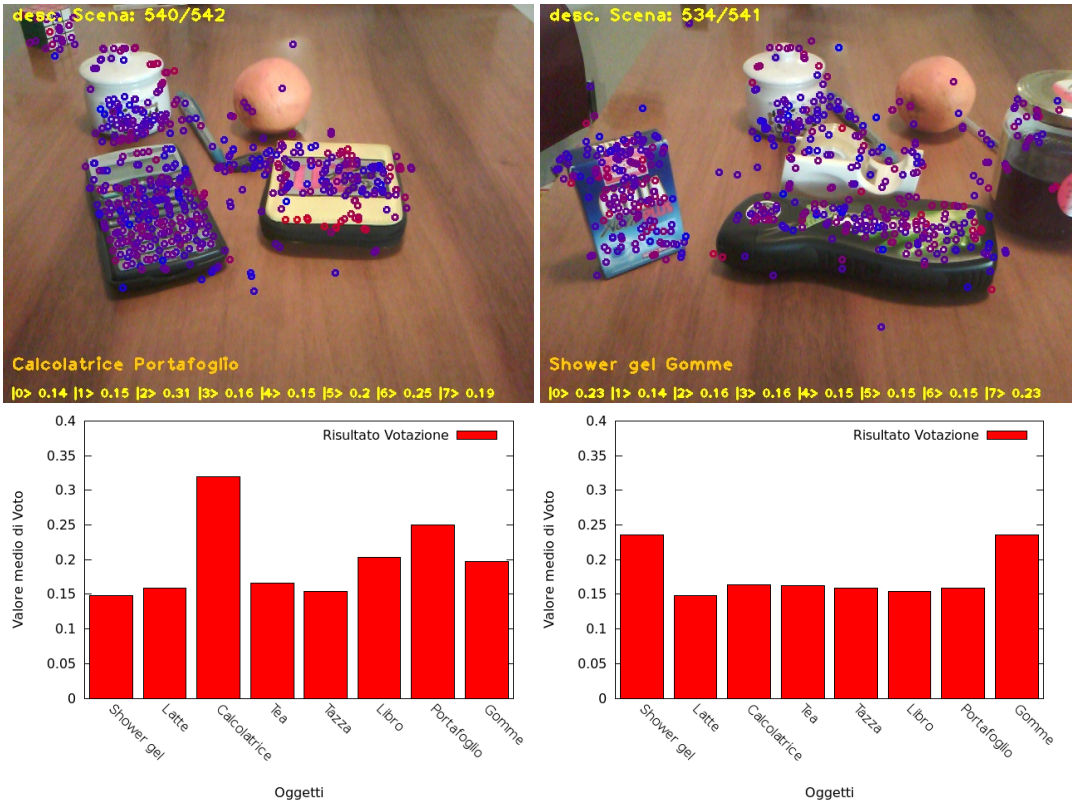


Figura 9: Matching multipli riusciti

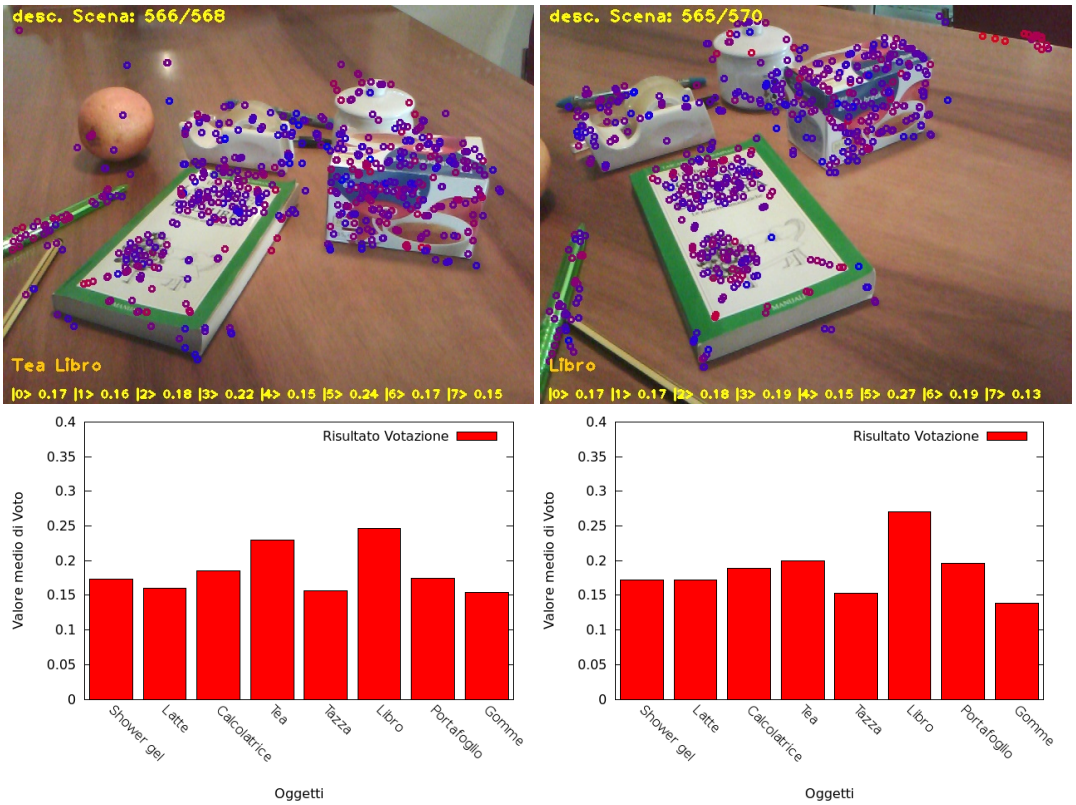


Figura 10: Matching multipli parzialmente riusciti

5.2.5 Test secondario di riconoscimento con occlusione parziale del soggetto

Sono state poi analizzate scene in cui un oggetto del database era parzialmente nascosto da altri oggetti estranei: l'algoritmo incontra difficoltà nella ricognizione dell'oggetto, e la riuscita dell'operazione dipende fortemente da che parti sono visibili. Nelle immagini sottostanti si può notare come una rotazione del punto di vista comporti un calo dei voti.

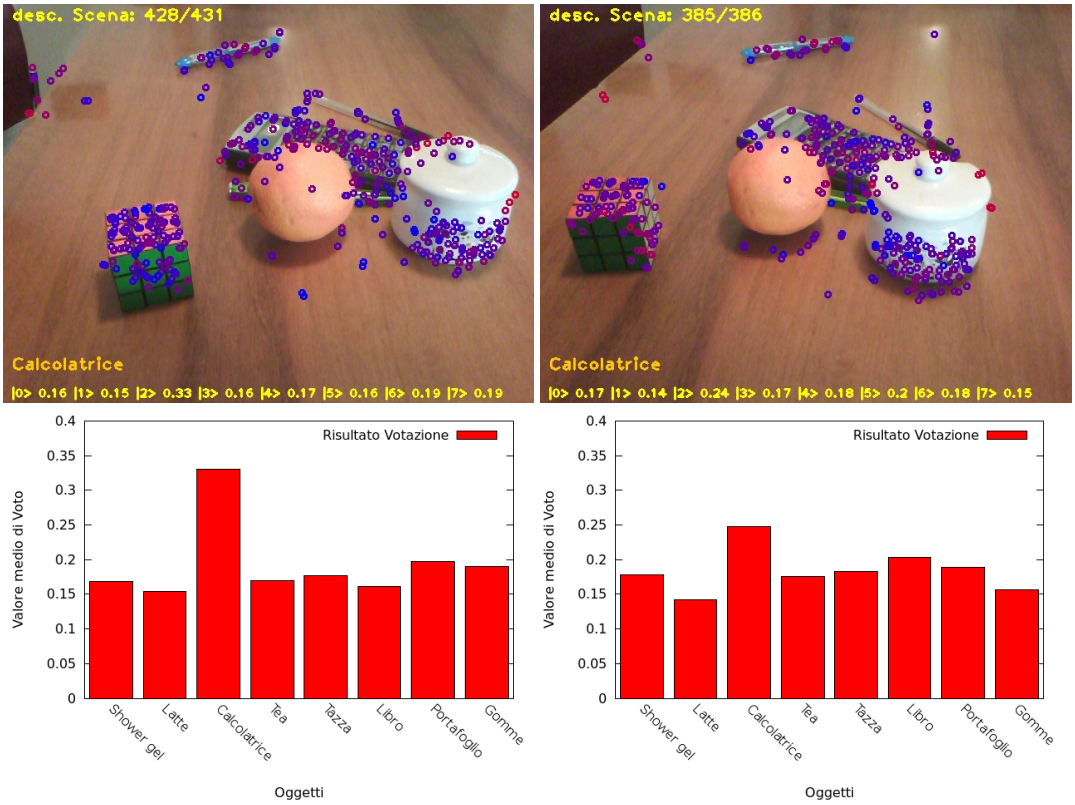


Figura 11: Matching con occlusione del bersaglio

5.2.6 Risultati dei test secondari

Nei test secondari l'algoritmo si comporta in maniera più instabile, variazioni di visuale, di luminosità e di sfondo cominciano a diventare determinanti per la riuscita o meno del rilevamento.

6 Conclusioni

6.1 Giudizio sulle funzionalità

Lo scopo principale, quello dell'efficienza, è stato raggiunto, difatti l'algoritmo è particolarmente veloce, e può essere eseguibile in real-time. Il programma si dimostra inoltre robusto nella ricognizione dei singoli oggetti e, anche se in misura minore, risponde bene al riconoscimento multiplo. In definitiva si pensa che il programma sia adatto ad essere montato sul robot, e che presenti una buona robustezza nel rilevamento degli oggetti pur rimanendo un algoritmo veloce e leggero.

6.2 Sviluppi futuri

L'implementazione di un sistema sliding windows rappresenta sicuramente un valido sviluppo. Aumenterebbe di sicuro la robustezza nel caso di oggetti multipli in scena, e nel contempo, permetterebbe di rilevare la zona dell'immagine dove si trova l'oggetto, un informazione utile al robot per sapere con precisione in che direzione spostarsi per la ricerca. Un ulteriore sviluppo dell'algoritmo potrebbe essere l'implementazione di un sistema di controllo sullo scarto quadratico medio dei voti in fase di apprendimento, in modo da avvertire l'utilizzatore che l'oggetto non è stato sufficientemente caratterizzato.

Riferimenti bibliografici

[1] C. Rother, V. Kolmogorov, A. Blake “GrabCut - Interactive Foreground Extraction using Iterated Graph Cuts”

[2] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool “SURF: Speeded Up Robust Features” (2007)

[3] Viola, P., Jones, M. ”Robust real-time object detection” Second International Workshop on Statistical and Computational Theories of Vision, (2001)

[4] Sivic, J. and Zisserman A. “Video google: A text retrieval approach to object matching in videos”, Proceedings of the International Conference on Computer Vision (2003)

[5] Nister, D. and Stewenius, H. “Scalable Recognition with a Vocabulary Tree”, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2006)

[6] Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A. “Object retrieval with large vocabularies and fast spatial matching”, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2007)

[7] Marszalek, M., Schmid, C. “Spatial weighting for bag-of-features”, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2006)

[8] Winn, J., Criminisi, A., Minka, T. “Object categorization by learned universal visual dictionary”, Proceedings of the International Conference on Computer Vision (2005)

[9] Leibe, B., Micolajckzyk, K., Schiele, B. “Efficient clustering and matching for object class recognition”, Proceedings of British Machine Vision Conference (BMVC’06) (2006)

[10] Lazebnik, S., Raginsky, M. “Learning nearest-neighbor quantizers from labeled data by information loss minimization”, Proceedings of Conference on Artificial Intelligence and Statistics (2007)

[11] Sito internet del progetto ROS “`http://www.ros.org/wiki/`”

[12] David G. Lowe “Object Recognition from Local Scale-Invariant Features ”, Computer Science Department University of British Columbia

[13] J. B. MacQueen ”Some Methods for classification and Analysis of Multivariate Observations”, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability (1967)

[14] Sito internet OpenCV “`http://opencv.willowgarage.com/wiki/`”

[15] Sito internet Gnuplot “`http://www.gnuplot.info/`”

Appendice A - Richiami di Teoria delle Probabilità

Si richiamano di seguito alcune nozioni di teoria della probabilità utilizzate nei precedenti capitoli.

Probabilità condizionata

Definizione di probabilità condizionata

Sia dato un evento C possibile e di probabilità strettamente positiva. Per ogni evento A si definisce la probabilità condizionata di A rispetto all'evento condizione C come:

$$p(A|C) = \frac{p(A,C)}{p(C)}$$

Proprietà notevole

La probabilità congiunta di due eventi A e B condizionati da C si scrive:

$$p(A,B|C) = p(A|B,C) \cdot p(B|C)$$

Regola di Bayes

Regola di Bayes

Dati due eventi di probabilità positiva B e C vale:

$$p(C|B) = \frac{p(B|C) \cdot p(C)}{p(B)}$$