UNIVERSITÀ DEGLI STUDI DI PALERMO

FACOLTÀ DI INGEGNERIA

Corso di Laurea Specialistica in Ingegneria Informatica

STUDIO E IMPLEMENTAZIONE DI TECNICHE PER IL RICONOSCIMENTO IN TEMPO REALE DELLO STATO DELLA MANO, BASATE SU RETE NEURALE, A PARTIRE DA OSSERVAZIONI MEDIANTE SENSORE MICROSOFT KINECT

TESI DI LAUREA DI

VITO GENTILE

RELATORE

PROF. ING. ANTONIO GENTILE

CONTRORELATORE

DOTT. GIORGIO VASSALLO

CORRELATORE

ING. SALVATORE SORCE

ANNO ACCADEMICO 2012 - 2013

Sommario

1. Introduzione

La diffusione del sensore Microsoft Kinect, e il vantaggio di poterlo sfruttare a costi molto limitati, ha consentito l'abbattimento di molte problematiche che impedivano un rapido avanzamento della ricerca. Con Microsoft Kinect, infatti, ogni utente può avere facilmente accesso a un'immagine a colori, sincronizzata con dati sulla profondità degli oggetti ripresi. Insieme a queste informazioni, Kinect fornisce uno strumento di riconoscimento di persone nella scena, fornendo molto rapidamente la posizione di alcuni punti salienti degli scheletri identificati. Utilizzando tutti questi dati, Microsoft, così come molti sviluppatori e ricercatori, hanno messo a punto numerose librerie per il riconoscimento di gesti del corpo.

Una carenza del sensore Kinect, tuttavia, è la forte limitazione esistente sulle risoluzioni sia delle immagini a colori, sia delle informazioni di profondità: per poter lavorare a 30 fps, infatti, bisogna limitarsi ad una risoluzione di 640x480 pixel. Ciò comporta che una persona a qualche metro di distanza dal sensore, sarà visualizzata in un'area relativamente piccola delle immagini. La situazione peggiora ulteriormente se si vuole ragionare sullo stato delle mani di un utente.

In questo lavoro sarà presentato un metodo che consente di riconoscere se le mani dell'utente sono chiuse o meno, supponendo che non ci siano oggetti tra il sensore e l'utente, e che quest'ultimo stia in piedi di fronte al sensore ad una distanza compresa tra 1,5 e 2,5 metri. Questo range di distanze consente, tra l'altro, l'integrazione del sistema qui presentato con altri sistemi di riconoscimento dei gesti del corpo, e questo può generalmente rappresentare un vantaggio in ambito applicativo.

Il metodo descritto utilizza i dati sensoriali per la segmentazione della mano, sfruttando sia le informazioni di profondità, sia quelle relative allo scheletro; esso inoltre si basa su considerazioni antropometriche per dimensionare alcune grandezze. Una volta estratta una regione di interesse contenente informazioni sulla sola mano, essa viene processata ed utilizzata come input di una rete neurale che funge da classificatore. L'output della rete sarà quindi filtrato con un processo di media temporale, al fine di ridurre il rumore. L'intero metodo, che si presta ad applicazioni

real-time, sarà infine analizzato sperimentalmente, per valutarne la bontà e la robustezza.

Il lavoro di tesi è stato accettato per la pubblicazione nel seguente lavoro: S. Sorce, V. Gentile, A. Gentile, “Real-time hand pose recognition based on a neural network using Microsoft Kinect”, in: Proceedings of the 8th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA 2013), Compiegne (France), 28-30 Ottobre 2013, IEEE Press.

2. Il sensore Microsoft Kinect

Kinect è un dispositivo hardware prodotto da Microsoft ed in vendita da novembre 2010, originariamente nato al fine di aumentare le possibilità di gioco della console Xbox 360. L'idea è quella di dotare un giocatore delle funzioni generalmente fornite da un controller di gioco, senza la necessità di utilizzare strumenti hardware aggiuntivi (quali game pad o simili). Per questo motivo, spesso si utilizza la locuzione "controller umano" per indicare ogni giocatore o utente ripreso dal Kinect.

L'uscita di Kinect ha suscitato grande interesse non solo in ambito ludico, ma anche tra gli sviluppatori software e nelle comunità scientifiche (in particolare nei settori della robotica, della realtà virtuale e del riconoscimento dei gesti). Da febbraio 2012, inoltre, Microsoft ha reso disponibile una versione di Kinect per PC dotati di Windows 7 e Windows 8, ampliando ulteriormente il campo applicativo. In quest'ultima versione del dispositivo, inoltre, sono state rese fruibili alcune funzionalità aggiuntive, quali la possibilità di modificare il range di distanze misurabili, diverse impostazioni delle fotocamere, ed altro ancora [1].

2.1. Descrizione dell'Hardware

L'hardware di Microsoft Kinect si basa su una tecnologia inizialmente studiata e messa a punto da 3DV, una compagnia che Microsoft ha prima finanziato e poi acquistato nel 2009, nonchè sul lavoro dell'azienda israeliana Prime Sense, che ha poi concesso in licenza i suoi brevetti a Microsoft.

Il 13 maggio 2010 è stato pubblicato negli U.S.A., proprio da Prime Sense, un brevetto che spiega esattamente la tecnologia implementata in Kinect [2].

Il dispositivo, dopo un disassemblaggio, si presenta nel modo seguente:

Figura 1. Disassemblaggio del Kinect (tratto da [3]).

L'apertura del Kinect mostra la presenza dei seguenti dispositivi:

- una videocamera a colori RGB, dalla quale è possibile estrarre uno stream di dati RGB con 8 bit di profondità, ad una risoluzione di default di 640x480 pixel. Con queste impostazioni, il frame rate è di 30 fps. Sono anche disponibili risoluzioni fino a 1280x1024 (a frame rate più bassi), e spazi di colore alternativi (come YUV);
- un sensore di profondità, costituito da un proiettore laser ad infrarossi combinato con un sensore monocromatico CMOS, sensibile alla stessa banda dei raggi proiettati. Da esso è possibile estrarre uno stream di dati di profondità ad 11 bit, cioè con 2048 livelli di profondità. Il range di profondità che Kinect può riconoscere varia tra 0,8 e 4 metri circa[1], sebbene, per consentire il corretto riconoscimento del corpo e dei gesti dell'utente, Microsoft suggerisce di lavorare entro un range di 1,2 - 3,5 metri [4];
- un array di microfoni (microfono stereo), che può essere utilizzato al fine di riconoscere comandi vocali, e che offre la possibilità di interpretare i suoni in funzione della direzione di provenienza; esso è anche utilizzato per la

[1] In realtà, la versione di Microsoft Kinect per Windows può lavorare anche in "Near Range Mode", cioè con un range di profondità pari a 0,4-3 metri [4].

calibrazione del sistema mediante l'analisi della riflessione del suono sulle pareti e sull'arredamento. In tal modo, il rumore di fondo viene eliminato, garantendo una maggiore capacità di riconoscimento vocale;

- una ventola per la dissipazione del calore;
- 64 MB di memoria flash DDR2;
- un accelerometro Kionix MEMS KXSD9 a tre assi, utilizzato per controllare l'inclinazione e per stabilizzare l'immagine;
- Prime Sense PS1080-A2, il chip che rappresenta il cuore della tecnologia Kinect.

E' inoltre presente un motore in grado di modificare l'angolazione della barra orizzontale del Kinect rispetto al piano su cui giace il dispositivo; esso può essere utilizzato anche durante il funzionamento degli altri sensori.

Figura 2. Principali dispositivi inclusi in Microsoft Kinect.

Sfruttando le informazioni prodotte dai sensori, Kinect è in grado di riconoscere simultaneamente fino a quattro persone, sia in piedi che seduti, estraendo, per ognuna di esse, fino a 20 giunti, che ne identificano lo scheletro. Ad ognuno di questi giunti viene assegnata una posizione in termini di distanza, nelle tre direzioni, dal dispositivo Kinect.

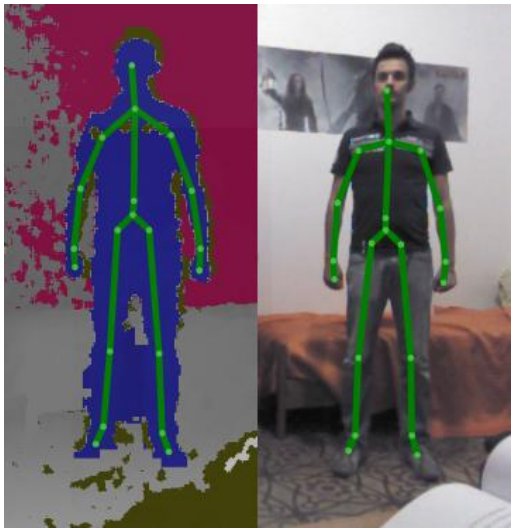


Figura 3. Due immagini ottenute (a breve distanza temporale l'una dall'altra) mediante Kinect. A sinistra, un esempio di immagine di profondità, in cui è evidente la rilevanza del rumore. A destra, l'immagine RGB. In entrambi i casi sono evidenziati i venti giunti che costituiscono lo scheletro estratto.

Visti i componenti hardware, uno schema di funzionamento del dispositivo potrebbe essere il seguente:

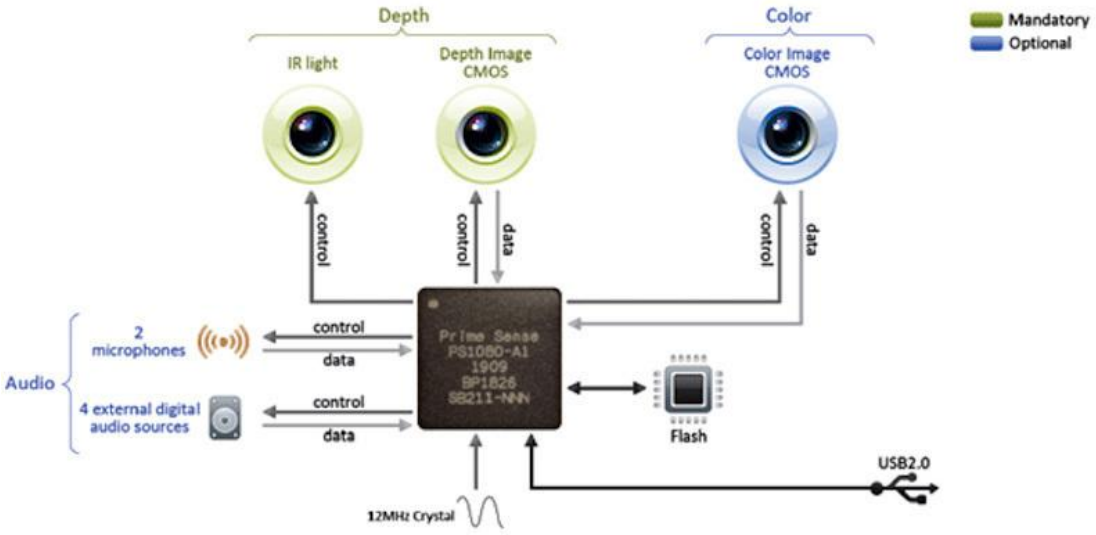


Figura 4. Schema di funzionamento del Kinect (tratto da [3])

La figura evidenzia come il chip PS1080-A2 di Prime Sense sovraintenda tutta la procedura di analisi della scena, controllando adeguatamente gli apparati ottici e audio, al fine di raccogliere le informazioni di cui necessita.

2.2. Funzionamento del sensore di profondità

Il sensore Microsoft Kinect ottiene informazioni 3D calcolando una mappa di disparità della scena. Normalmente, questo genere di mappe viene calcolato per derivare la posizione 3D di ogni punto di una scena tramite un sistema di stereovisione. Quest'ultima è una tecnica di ottica inversa che consiste nell'ottenere informazioni di profondità da una coppia di immagini provenienti da due telecamere che inquadrano la stessa scena da posizioni diverse.

Supponendo di avere a disposizione un sistema di stereovisione costituito da due telecamere RGB leggermente distanziate ed ugualmente orientate, potremmo ottenere una mappa di disparità confrontando coppie di pixel delle due immagini corrispondenti allo stesso punto della scena. Dal momento che il Kinect è dotato solo di una videocamera RGB ed un sensore di profondità (costituito a sua volta da un proiettore infrarossi ed una fotocamere sensibile a tale banda), la soluzione adottata è la seguente: l'emettitore infrarossi proietta nell'ambiente esaminato un pattern ben definito di spot luminosi (grazie ad una mascherina posta davanti al proiettore). La disposizione dei punti del pattern è nota a priori al Kinect, che al suo interno ha memorizzato come il pattern dovrebbe essere visto dal sensore infrarossi se fosse proiettato su una superficie posta ad una distanza ben definita e perfettamente parallela al piano della depth camera.

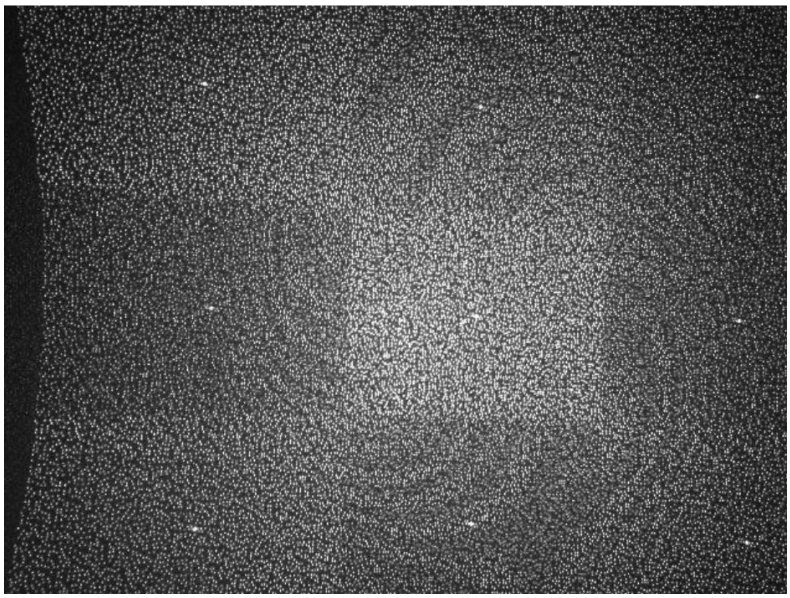
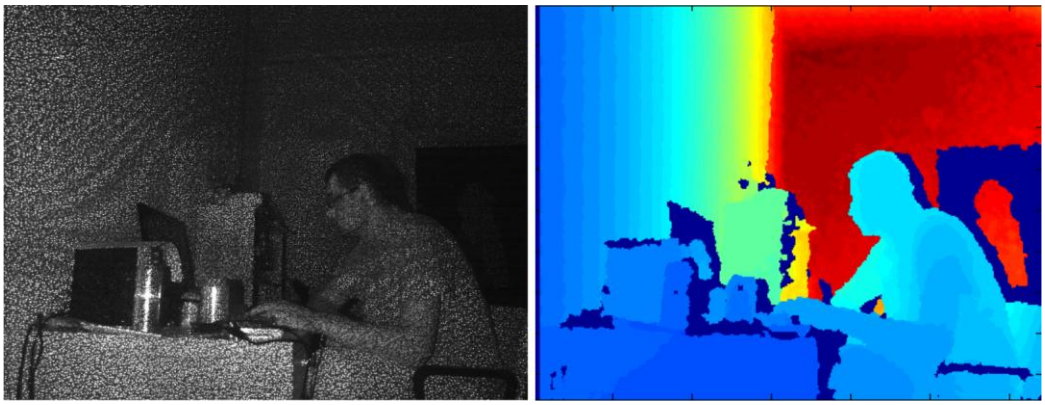


Figura 5. Esempio di come il pattern IR verrebbe visualizzato su una parete (tratto da [5]).

Confrontando la distribuzione dei punti proiettati con il pattern di riferimento memorizzato all'interno del Kinect, si può ottenere una mappa di disparità mettendo in relazione la posizione dei punti del pattern osservato, con quella dei punti del pattern di riferimento, tramite considerazioni geometriche che esulano dalle finalità del presente elaborato. Per utilizzare un esempio chiarificatore, si può pensare al pattern di riferimento come ad una coperta a pois, parallela al piano della depth camera e posta ad una distanza ben definita. Se un oggetto entra a contatto con la coperta, questa verrà deformata, e di conseguenza chiunque potrà intuire la disposizione spaziale dell'oggetto, sulla base della nuova disposizione dei pois della coperta.



(a) (b)

Figura 6. Confronto tra l'immagine acquisita dalla camera ad infrarossi (a), e la corrispondente immagine di profondità, qui rappresentata in falso colore (b) (tratto da [5]).

La funzione del chip Prime Sense PS1080-A2 è proprio quella di ricavare la mappa di disparità a partire dai dati sensoriali. La capacità di effettuare tali calcoli in hardware è il punto di forza di Microsoft Kinect.

2.3. Tracking dello scheletro

Una volta ottenuta l'immagine di profondità della scena osservata, il successivo step di elaborazione consiste in un'operazione di tracking interamente a carico del software: l'identificazione del numero, della posizione e delle giunture scheletriche degli esseri umani presenti all'interno della scena. Con il termine tracking si indica la capacità di un calcolatore di riconoscere la posizione e l'orientamento di determinati oggetti attraverso l'analisi di una o più immagini in

sequenza. Nel caso del sensore Kinect, l'analisi riguarda un singolo frame, ed il riconoscimento è relativo agli esseri umani presenti nella scena.

Il dispositivo Kinect effettua la cosiddetta "pose recognition in parts" [6] [7], ovvero la predizione delle posizioni 3D ed il tracciamento delle giunture scheletriche del corpo umano, a partire da una singola immagine di profondità. L'approccio si rifà alle moderne tecniche nell'ambito dell'object recognition, basate sul principio della suddivisione degli oggetti in parti. In questo caso, infatti, è il corpo umano ad essere suddiviso in porzioni, rappresentanti le parti del corpo (mani, testa, braccia, gambe, etc...).

L'unico input al software è costituito dalla depth image (in scala di grigi), prodotta dal chip Prime Sense, dalla quale si provvede ad eliminare qualsiasi elemento di sfondo, al fine di isolare la fisionomia del controller umano. Su tale immagine viene operata un'etichettatura probabilistica, che assegna ad ogni punto dell'immagine di profondità in input, le probabilità che tale punto appartenga ad ogni porzione del corpo. In tal modo, verranno generate tante distribuzioni di probabilità quante sono le parti del corpo prese in esame.

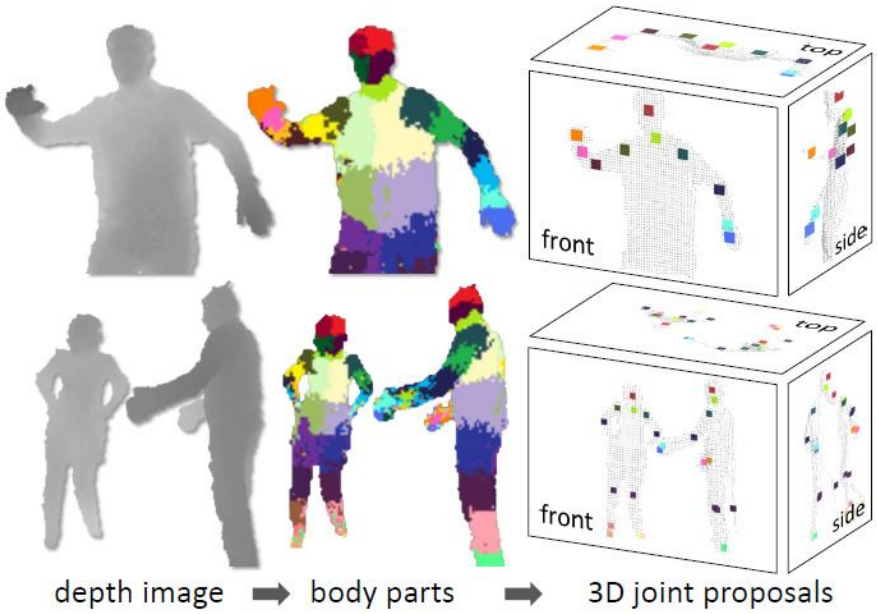


Figura 7. Processo di etichettatura dell'immagine di profondità, seguito dall'ottenimento dei giunti scheletrici del corpo umano ripreso (tratto da [6]).

Questo tipo di rappresentazione trasforma il complesso problema del tracking, in uno facilmente risolvibile con opportuni algoritmi di classificazione, che

associano ogni punto dell'immagine di profondità a determinate classi, rappresentanti le parti del corpo. Il processo di classificazione è effettuato utilizzando un algoritmo basato sull'addestramento di alberi decisionali. Senza scendere nei dettagli dell'algoritmo, si precisa che l'insieme di addestramento è stato generato sfruttando alcuni modelli 3D generati artificialmente, dai quali può essere ricavata sia un'immagine di profondità (input), sia la corrispondente etichettatura probabilistica (output atteso), in situazioni tipiche e particolarmente significative per garantire un'apprendimento quanto più soddisfacente. In particolare, Microsoft ha ricavato un database di 500.000 frame, dai quali è stato ottenuto un training set di 10.000 esempi significativi per la classificazione. Le immagini seguenti mostrano alcuni modelli 3D utilizzabili per ottenere il training set appena citato.

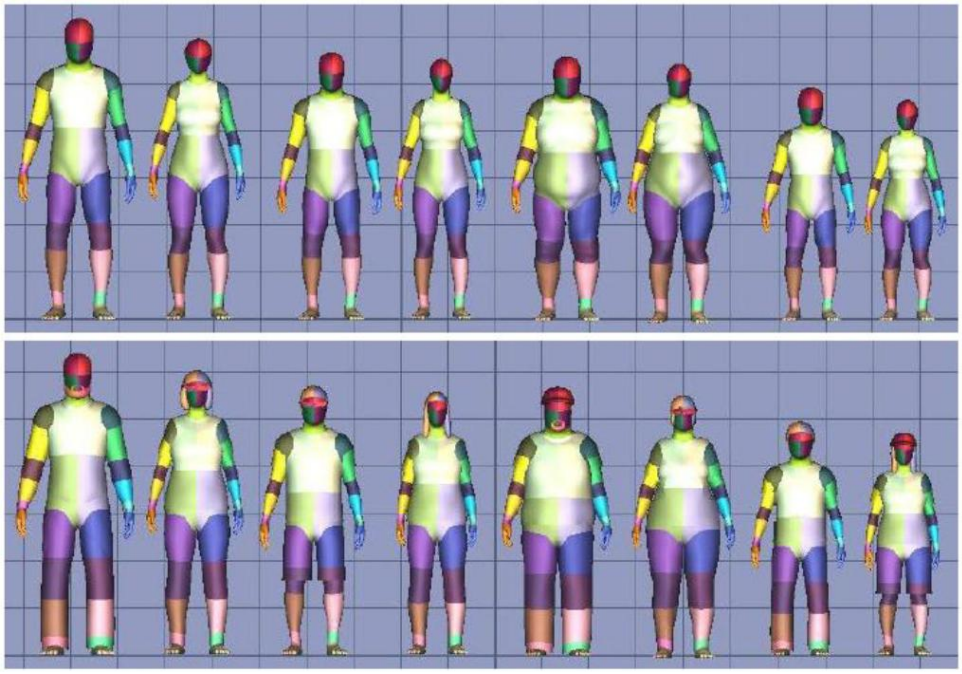


Figura 8. Esempi di modelli 3D dai quali è possibile ricavare esempi per generare un insieme di addestramento (tratto da [7]).

(a)

(b)

Figura 9. Confronto tra (a) alcuni elementi del training set, generati artificialmente, e (b) alcuni esperimenti reali, ottenuti tramite l'algoritmo di pose recognition in parts (immagini tratte da [6]).

Una volta ottenuta l'etichettatura (che può essere pensata come una texture map da applicare ai modelli 3D, in cui ogni colore identifica una parte del corpo), si ricavano i singoli punti che rappresentano i giunti scheletrici. Essi vengono calcolati tramite un algoritmo di ricerca della moda di ciascuna distribuzione di probabilità (ognuna corrispondente ad una parte del corpo).

I ricercatori Microsoft che si sono occupati dello sviluppo dell'intero algoritmo di human pose recognition in parts, affermano che un'implementazione ottimizzata di questo algoritmo permette l'analisi di un frame e l'estrazione dello scheletro in circa 5 ms, utilizzando la GPU integrato in Xbox 360 [6]. Tale tempo è sufficiente a garantire un funzionamento in real time.

L'immagine seguente mostra come i risultati del processo di tracking si rivelino soddisfacenti in una situazione tipica e, al tempo stesso, relativamente significativa.

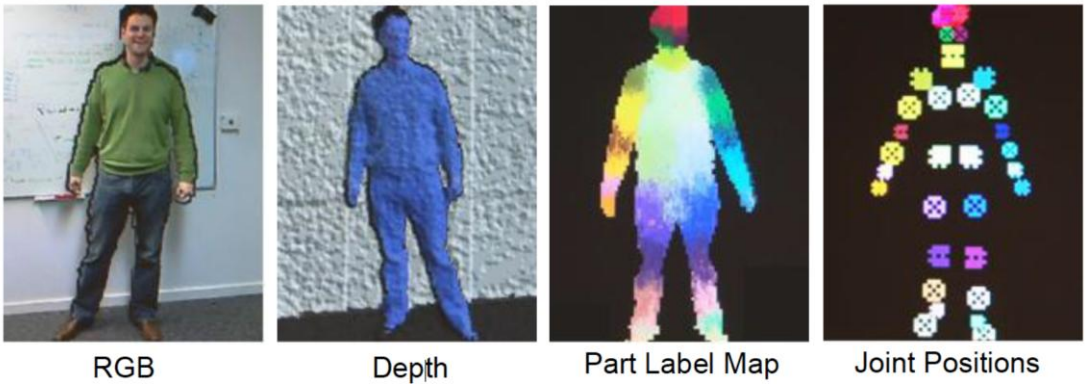


Figura 10. Esempio di funzionamento dell'algoritmo di tracking (tratto da [5]).

2.4. Strumenti software

Dopo il rilascio del dispositivo sul mercato, Microsoft non ha reso subito disponibili i driver per Windows o altri sistemi operativi, rischiando di rallentare inizialmente lo sviluppo. A dicembre 2010, però, la società Prime Sense ha rilasciato un pacchetto di driver open source per Kinect, compatibili con Windows e Linux, basati su un'API completa nota come OpenNI (Open Natural Interactions) [8]. Inizialmente, quindi, era possibile sfruttare questo sistema di librerie per programmare software per Kinect.

Nel giugno 2011, Microsoft ha rilasciato i driver ufficiali, con licenza non commerciale. Insieme ad essi, viene fornito un SDK ed un framework che consente, grazie anche all'integrazione con Microsoft Visual Studio, di facilitare lo sviluppo di applicazioni che utilizzano Kinect. Sono, inoltre, supportati vari linguaggi di programmazione, quali C#, C++ e Visual Basic 2010.

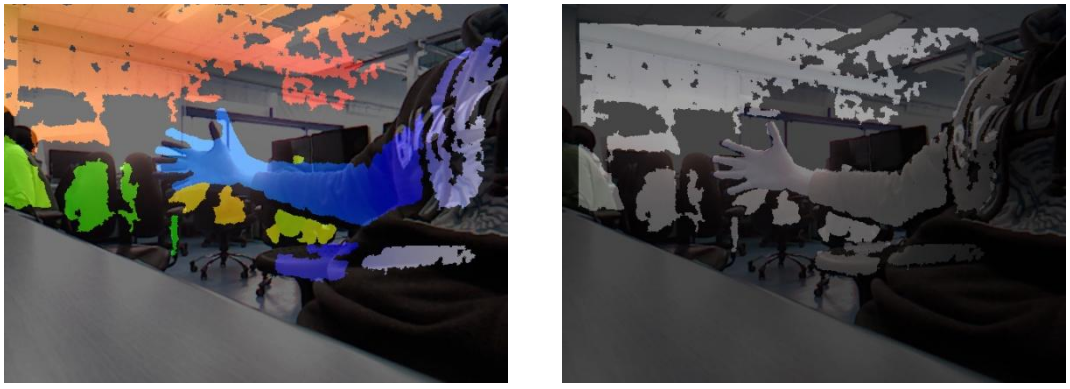
In congiunzione agli strumenti software forniti da Microsoft, è bene sottolineare l'importanza di alcune librerie fondamentali per lo sviluppo di applicazioni per Kinect. In generale, infatti, si ha a che fare con concetti relativi all'elaborazione delle immagini digitali. A tale scopo, gli sviluppatori C++ possono sfruttare OpenCV [9], una delle più diffuse librerie software orientata alla visione artificiale, e progettata per essere efficiente e sfruttare al meglio le possibilità offerte dalla programmazione parallela, eventualmente utilizzando anche le GPU. Poichè questa soluzione non è disponibile per C#, è stata sviluppata un'altra libreria open source, chiamata Emgu CV [10]: si tratta di un wrapper di OpenCV in grado di

fornire tutte le sue funzionalità anche agli sviluppatori .NET. Nell'implementazione del sistema descritto nel seguito, si è fatto uso di Emgu CV.

2.5. Limitazioni e punti di forza

L'unità centrale di Microsoft Kinect fornisce molte funzionalità interessanti, come la capacità di estrarre le informazioni sullo scheletro in maniera estremamente veloce. Ci sono, tuttavia, alcune limitazioni che è bene citare, al fine di comprendere meglio le motivazioni di alcune scelte intraprese nel seguito.

La videocamera RGB ed il sensore di profondità, innanzitutto, non sono (e non possono essere) perfettamente allineati. Ciò implica che le immagini RGB e le informazioni di profondità devono essere sottoposte ad un processo di registrazione per potere essere utilizzate con precisione. In aggiunta, se si rende necessario confrontare i punti dello scheletro con i suddetti dati sensoriali, sarà necessaria un'ulteriore operazione di mappatura dei giunti. Questo genere di problematiche, se non attenzionate, possono rallentare la computazione, poichè possono essere abbastanza onerose. In ogni caso, l'SDK facilita le operazioni di mapping dei dati sensoriali mediante l'utilizzo di API appositamente studiate [11].



(a) (b)

Figura 11. Un esempio (tratto da [12]) che mostra la sovrapposizione di un'immagine di profondità con quella a colori, prima non correttamente registrate (a), e poi a seguito di un'opportuna operazione di mapping.

Un altro problema che limita notevolmente l'utilizzo di Microsoft Kinect è la ridotta risoluzione sia del sensore di profondità che della videocamera RGB. Al crescere della distanza dal sensore, infatti, il numero di pixel coinvolti per la

rappresentazione di un qualche oggetto diminuisce; questo è il motivo per il quale il limite per il riconoscimento dello scheletro è inferiore a 4 metri. Se si vuole effettuare una misurazione o un'elaborazione relativa ad oggetti più piccoli dell'intero corpo umano (come, per esempio, il riconoscimento di una mano), la distanza massima non può che essere ulteriormente inferiore. In aggiunta a queste considerazioni generali, prendendo in esame il solo caso di default, e cioè quello che utilizza una risoluzione di 640x480 pixels, uno studio di accuratezza del sensore di profondità di Microsoft Kinect, condotto da Khoshelham [13], ha dimostrato che l'andamento della deviazione standard dell'errore (approssimato ad una perturbazione gaussiana a media nulla) cresce quadraticamente con la distanza, arrivando a valere circa 4 cm per un utente posto a 4 metri dal sensore. Se si considerano grandezze dell'ordine del decimetro (come una mano umana), ridurre la distanza massima ammissibile è un vincolo obbligatorio.

3. Riconoscimento della mano: stato dell'arte

Il processo di riconoscimento dello stato della mano è stato studiato da molti autori, alcuni dei quali si sono specializzati in metodi che sfruttano le funzionalità di Microsoft Kinect. In generale, questi lavori presentano alcuni tratti comuni, che è bene riepilogare nel seguito.

Un primo problema che viene affrontato è quello che riguarda la localizzazione e la segmentazione della mano. I metodi per effettuare queste operazioni si basano sull'utilizzo sia dei dati di profondità, sia di quelli ottenuti dalle fotocamere RGB. Spesso, dopo una fase di individuazione di un punto centrale della mano[2], viene effettuata una sogliatura basata sui dati di profondità [14] [15] [16]; in alternativa, alcuni ricercatori preferiscono utilizzare una segmentazione basata sul colore della pelle, utilizzando tecniche che spaziano dalle più semplici e tradizionali (sogliatura di Otsu [17], utilizzata da Matos et al. [18]), a modelli probabilistici [19], fino all'utilizzo di spazi di colore più appropriati [20]. Alcuni ricercatori, inoltre, adoperano algoritmi di region growing basati su proprietà di colore e/o di profondità [20]. Non mancano autori che utilizzano approcci ibridi, cioè in grado di combinare la segmentazione (o il processo di region growing) basata sul colore della pelle, con i risultati ottenuti mediante le informazioni di profondità [20] [21].

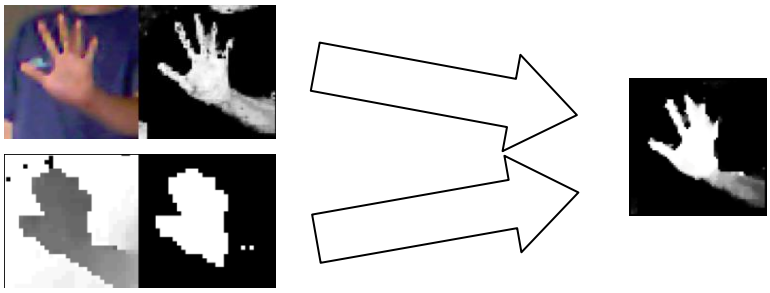


Figura 12. Un esempio di segmentazione ottenuta combinando le informazioni di profondità e colore (tratto da [21]).

[2] L'individuazione del punto della mano può essere effettuata anche dopo un processo di segmentazione. In questo caso, si potrebbe considerare il centroide della regione che è selezionata come rappresentativa della mano, e stabilire questo punto come riferimento per le successive elaborazioni.

Per perfezionare il risultato della segmentazione, in letteratura si utilizzano spesso operazioni di post-processing che semplificano le fasi successive. Un problema che si verifica frequentemente, ad esempio, è quello di separare la mano da parte dell'avambraccio (che spesso, essendo ad una profondità circa pari a quella del palmo della mano, viene incluso nei risultati della segmentazione per mezzo delle immagini di profondità). Al fine di risolvere questo tipo di problematica, sono stati proposti metodi di vario tipo; Ren et al. [22] sfruttano l'assuzione che l'utente indossi un braccialetto nero di separazione, e utilizzano quest'ultimo come edge di riferimento per la suddivisione; Xu et al. [23] e Bhuyan et al. [19], invece, effettuano la separazione delle due regioni basandosi su considerazioni geometriche. Altri problemi sono legati allo smoothing della forma della mano, che generalmente può essere effettuato con un semplice filtro mediano [24]; in altri casi, allo scopo di velocizzare la computazione, il contorno può essere approssimato con una curva polinomiale a tratti, eventualmente lineari [25].

(a) (b)

Figura 13. Due esempi di operazioni di post-processing, che seguono la segmentazione. La figura (a), tratta da [25], mostra i risultati dell'applicazione di un filtro mediano per l'arrotondamento del boundary; la figura (b), tratta da [23], schematizza il funzionamento di un algoritmo per la separazione della mano dall'avambraccio.

Una volta ottenuta l'immagine della mano, essa deve essere classificata in modo da stabilirne la configurazione (cioè in modo da stabilire se la mano è aperta, chiusa o eventualmente se assume una tra diverse altre possibilità). Per fare questo, in genere, si estraggono delle caratteristiche salienti dall'immagine segmentata e le si utilizzano come input di un classificatore opportuno. Ahmed [26] e Biswas et al. [14] suddividono l'immagine segmentata in diverse zone, e da ognuna di queste ricavano alcuni valori da utilizzare come features; Matos et al. [18] ricavano lo scheletro della mano, utilizzando poi alcune informazioni ricavate da esso; Poonam et al. [27] e Bhuyan et al. [19] ottengono delle features basate su caratteristiche geometriche. In alternativa a ciò, si può rappresentare la forma della mano utilizzando una signature

del contorno [22]. Le features più interessanti, probabilmente, sono quelle che offrono invarianza ad operazioni quali la scalatura e la rotazione; questo genere di requisito è stato considerato da Matos et al. [18], che normalizza la rotazione della mano prima di ricavarne lo scheletro, e da Bagdanov et al. [15], che sfrutta i descrittori SURF [28] [29], avvalendosi delle loro proprietà di invarianza alla scalatura ed alla rotazione.

Per la classificazione, le tecniche più adoperate sono quelle che utilizzano reti neurali [30] o macchine a vettore di supporto (Support Vector Machine, SVM), che richiedono una prima fase di addestramento. Questa operazione richiede la raccolta di grandi moli di dati di esempio, che generalmente non è semplice (sebbene molti autori propongono alcuni training set da loro messi a punto [22] [15]). In alternativa, seguendo l'approccio di Ren et al. [22], si può procedere al calcolo di una opportuna misura di distanza tra la mano segmentata (in una qualche rappresentazione conveniente) ed alcuni prototipi, scegliendo quella che ha la minore dissimilarità.

Un'ultima operazione di filtraggio può essere effettuata sull'uscita del classificatore, riducendone il rumore. Questo è possibile poichè è abbastanza improbabile che, tra un istante ed il successivo, lo stato della mano vari sensibilmente. Di conseguenza, si può ridurre il rumore dell'uscita con uno smoothing temporale. Alcuni autori suggeriscono, in alternativa, di utilizzare un filtro di Kalman (assumendo bianco il rumore dell'uscita) [15] [16].

Buona parte degli studi citati fino a questo punto riguardano il riconoscimento dello stato della mano supponendo che l'utente sia sufficientemente vicino al sensore Kinect, in modo da potere classificare la mano sulla base dello stato delle singole dita. Se, invece, si rimuove questa assunzione, non si può più fare affidamento sulle caratteristiche geometriche della mano, poichè la distinzione delle dita diventa impraticabile a causa della bassa risoluzione del sensore Kinect. Per questo motivo, molti dei metodi che si basano sul riconoscimento delle dita o di punti salienti della mano, nel seguito, non verrano presi in considerazione.

4. Algoritmo per il Riconoscimento

La discussione che segue procederà facendo riferimento al riconoscimento di una sola mano, dal momento che l'intero processo può facilmente essere esteso anche al riconoscimento simultaneo di entrambe le mani.

In aggiunta a quanto detto, tutte le grandezze ottenute dal sensore Kinect (profondità, posizione di punti, ecc...) possono essere espresse sia in metri che in pixel, grazie alle funzionalità di conversione e mapping di coordinate implementate da Microsoft e fornite con l'SDK. Per questi motivi ci si riferirà a queste misure in maniera indistinta, senza cioè esplicitare se si tratta di pixel o di metri. Sarà il contesto a fugare ogni dubbio.

4.1. Posizionamento dell'utente per il riconoscimento

Per consentire l'integrazione del sistema con altri algoritmi di riconoscimento dei gesti, il sensore Kinect deve essere in grado di visualizzare il corpo dell'utente nella sua interezza, anche durante l'esecuzione del gesto. Come già accennato, Microsoft suggerisce una distanza dal sensore compresa tra 1,2 e 3,5 metri per ottenere buone performance durante le operazioni di riconoscimento dei gesti [4]. Questo range di distanze, tuttavia, non può essere utilizzato per la specifica applicazione che si sta descrivendo. Si fornisce nel seguito una spiegazione.

Se si considerano alcuni dati antropometrici raccolti su una vasta scala di popolazione adulta [31] [32], uniti all'esperienza comune, si può concludere che, mediamente, una mano di un uomo o di una donna adulta rientra all'interno di un quadrato di lato pari a 25 centimetri.

Tuttavia, sulla base dello studio di Kholshelham [13] sull'accuratezza del sensore di profondità, ad una distanza di circa 4 metri, l'errore di misurazione è di circa 4 centimetri, che può risultare eccessivo per grandezze dell'ordine del decimetro. Per mantenere questo errore entro valori quanto più ragionevoli, la distanza massima a cui l'utente dovrebbe posizionarsi è fissata a 2,5 metri, in modo che l'errore sia limitato ad una perturbazione gaussiana con deviazione standard di circa 1 centimetro.

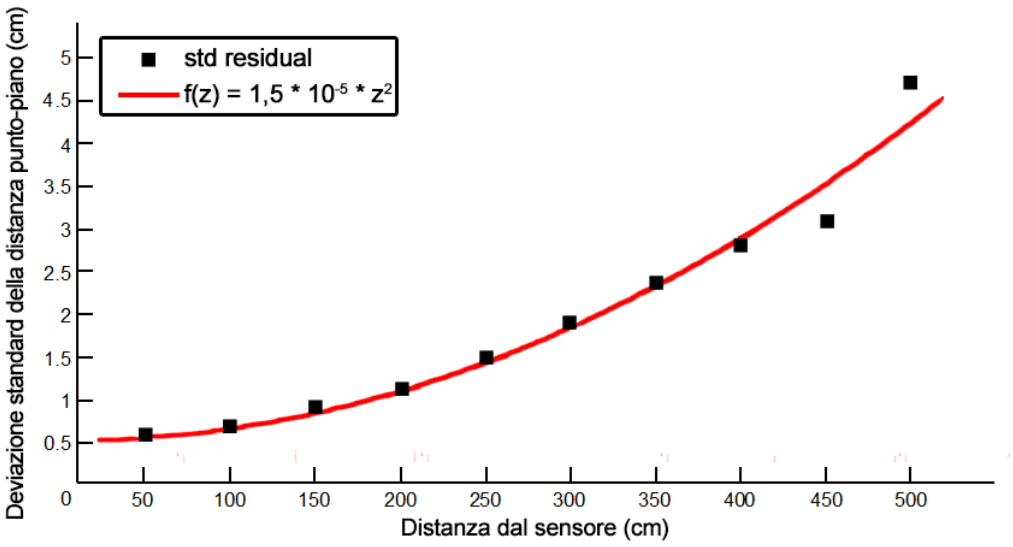


Figura 14. Andamento approssimativo della deviazione standard dell'errore gaussiano di misurazione [13]. Al crescedere dalla distanza dal sensore, la deviazione standard cresce quadraticamente. Per distanze minori di 2,5 metri, tuttavia, l'errore può essere considerato trascurabile.

Una volta stabilito il range di distanze ammissibili, che varia tra 1,5 e 2,5 metri, l'utente potrà posizionarsi in modo che la Kinect riconosca il suo scheletro. Sulla base di tutte le informazioni sensoriali ottenute dalla Kinect sarà quindi possibile utilizzare il metodo descritto nel seguito.

4.2. Avvio del processo di riconoscimento

Per prima cosa, è necessario stabilire in quali condizioni il sistema deve attivare la procedura di riconoscimento. E' inutile, infatti, che essa venga avviata se l'utente ha le braccia lungo il corpo, o se le ha dietro la schiena. Per stabilire se il processo possa essere attivato o meno, si verifica che la mano disti sufficientemente dal resto del corpo. La distanza limite può essere valutata sulla base delle informazioni relative allo scheletro, ed in particolare in funzione della lunghezza dell'avambraccio o del braccio intero. Per semplicità, si può anche decidere di fissare un valore che si rivela "sperimentalmente abbastanza buono". Il metodo qui descritto utilizza un valore di distanza pari a 15 cm come limite inferiore di distanza tra la mano e il centro di massa del corpo dell'utente[3].

[3] Le API forniscono un metodo per ottenere un singolo punto che identifica l'intero scheletro; si può ragionevolmente assumere che questo punto sia uguale al centro di massa dell'utente.

Sia dunque $P_H(x_H, y_H, z_H)$ il punto della mano nello spazio 3D, e sia $P_S(x_S, y_S, z_S)$ il centro di massa dell'intero utente, secondo il sistema di riferimento illustrato nella figura seguente:

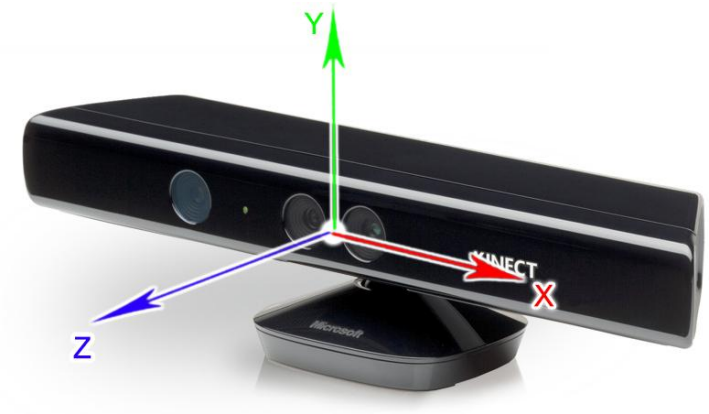
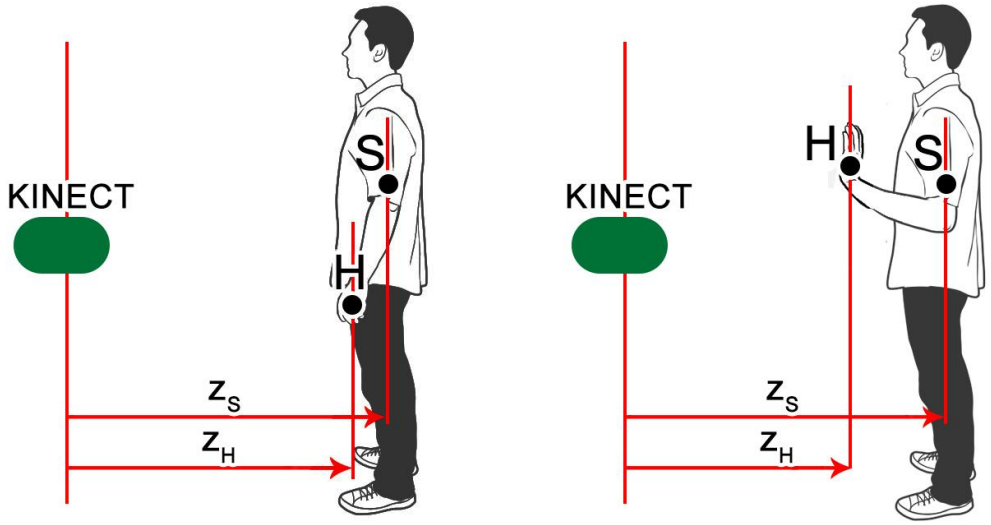


Figura 15. Il sistema di riferimento 3D utilizzato dal sensore Kinect

Affinchè possa essere avviato il riconoscimento, deve quindi valere che:

$$| z_H - z_S | > 0{,}15 \text{ m}$$

dove z_H e z_S si intendono espressi in metri, e rappresentano la distanza dei punti P_H e P_S dal sensore. Le immagini seguenti mostrano due esempi significativi che chiariscono il motivo della suddetta condizione.



(a) (b)

Figura 16. (a) Il braccio è lungo il corpo, e quindi la mano non è sufficientemente lontana dal centro di massa; di conseguenza, il processo di riconoscimento non può essere avviato. (b) La mano è sufficientemente distante dal corpo per far sì che il riconoscimento venga effettuato.

4.3. Procedura di Segmentazione della Mano

Sia ancora $P_H(x_H, y_H, z_H)$ il punto della mano nello spazio 3D ottenuto dallo scheletro riconosciuto dalla Kinect. Supponendo allineati i frame ottenuti dalla videocamera RGB e dal sensore di profondità[4], centriamo un quadrato di lato L pixels sul punto di coordinate (x_H, y_H). Ciò può essere fatto convertendo le informazioni di profondità in immagini in scala di grigio. Consideriamo quindi solo questo quadrato come contenitore della mano. Il lato L può essere scelto sulla base di esperimenti che assegnino ad L un valore "mediamente buono" per un bounding square della mano, oppure la scelta può basarsi su considerazioni antropometriche, che assegnino ad L un valore basato anche sulle dimensioni dell'intero corpo (cosa che può essere fatta misurando le distanze tra altri punti caratteristici dello scheletro). In questo contesto, si è utilizzato un valore di L pari a 25 centimetri. Cheng et al. [20] hanno stimato una relazione lineare tra la dimensione del palmo della mano in pixel e la profondità, ed un approccio di questo tipo potrebbe fornire risultati più precisi.

Siano Im_{depth} e Im_{Gray} le regioni di interesse che includono la mano, ricavate rispettivamente dall'immagine di output fornita dal sensore di profondità, e da quella ottenuta dal sensore RGB (quest'ultima convertita in scala di grigi). A questo punto, si effettua una binarizzazione dell'immagine di profondità Im_{depth}, ottenendo una nuova immagine binaria, che chiameremo Im_{mask}. La binarizzazione è ottenuta secondo la legge seguente:

$$Im_{mask}(x,y) = \begin{cases} BIANCO & se\ Im_{dept\,h}(x,y) \geq z_H - \Delta_{dept\,h} \\ NERO & altrimenti \end{cases}$$

[4] Come già accennato precedentemente, in realtà i dati RGB e di profondità non sono mai perfettamente allineati, a causa della distanza non nulla dei sensori montati su Microsoft Kinect. Tuttavia l'SDK è munito di procedure che effettuano il mapping in modo efficiente, ed è per questo che tale dettaglio può essere trascurato in questa sede.

(a) (b) (c)

Figura 17. Sfruttando l'informazione dello scheletro sulla posizione della mano, è possibile ottenere (a) un'immagine in scala di grigi della mano (Im_{Gray}), (b) l'informazione di profondità della mano (Im_{depth}) e utilizzare quest'ultima per ottenere (c) una maschera binaria della mano (Im_{mask}).

Qui il valore di Δ_{depth} rappresenta lo spessore della mano lungo la congiungente mano-sensore. Anche questo valore, in generale, deve essere stimato sulla base di considerazioni euristiche o, più propriamente, antropometriche. Negli esperimenti effettuati, considerata anche la necessità di un buon livello di tolleranza al rumore, si è visto che porre Δ_{depth} = 8 cm fornisce risultati accettabili.

(a) (b)

Figura 18. Illustrazione che spiega il significato geometrico di Δ_{depth}: (a) vista laterale; (b) vista dall'alto.

4.4. Input del Classificatore Neurale

La rete neurale utilizzata come classificatore (di cui si discuterà più accuratamente in seguito) dovrà avere come input le informazioni sensoriali relative alla mano. Sono stati testati tre diversi approcci, che saranno elencati e descritti brevemente nel seguito.

4.4.1. Maschere di Profondità

Un primo e più semplice approccio consiste nell'utilizzare la sola immagine binaria della mano, riportarla ad una dimensione fissata (nel caso in esame,

100x100), ed utilizzarla come input della rete neurale. Piuttosto che immettere in ingresso 10.000 valori binari, si è adottato un approccio basato sul calcolo degli istogrammi di proiezione verticale ed orizzontale. Da una immagine di dimensione LxL, si ottengono:

- Un vettore L-dimensionale, il cui i-esimo elemento rappresenta il numero di pixel bianchi dell'i-esima riga;
- Un vettore L-dimensionale, il cui i-esimo elemento rappresenta il numero di pixel bianchi dell'i-esima colonna.

Concatenando questi due vettori se ne ottiene uno solo di dimensione 2L, che nel caso di studio è pari a 200 elementi (contro 10.000 se si utilizzasse l'intera immagine linearizzata come input della rete).

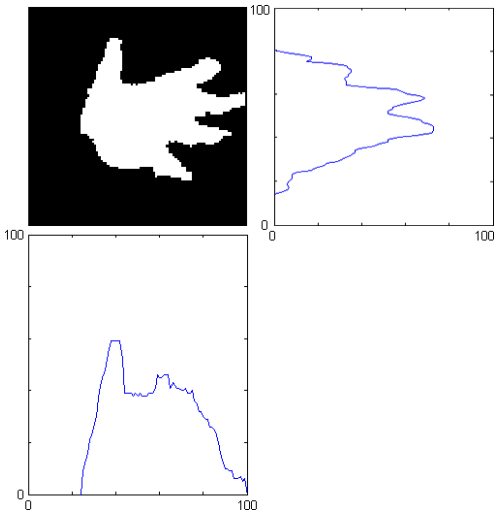


Figura 19. Gli istogrammi di proiezione verticale ed orizzontale della maschera binaria

Il problema di questo approccio può sorgere nel caso in cui si vuole tollerare che l'utente possa inclinare la mano in modo da renderla quasi parallela alla congiungente mano-sensore. In tal caso, la sola maschera binaria può essere troppo poco discriminante, e devono essere introdotti meccanismi più sofisticati. Un grande vantaggio, invece, è quello di non dover trattare l'immagine a colori, consentendo quindi l'utilizzo anche in condizioni di illuminazione poco vantaggiose, o nel caso in cui l'utente indossi guanti.

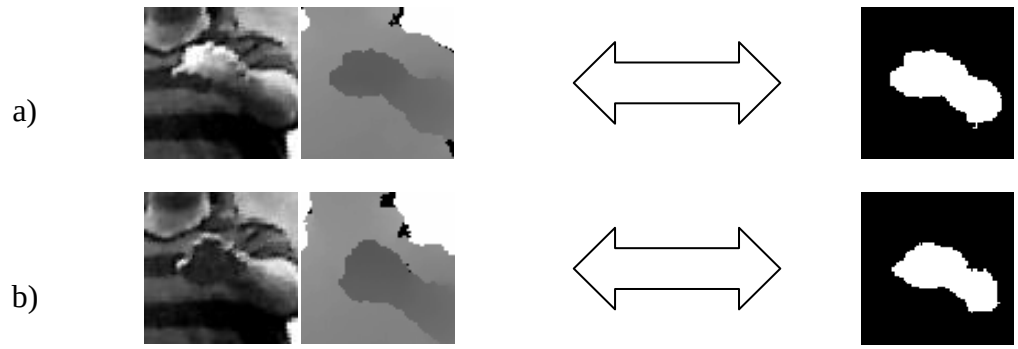


Figura 20. Le due immagini in scala di grigio mostrano le differenze tra (a) e (b), mentre le due maschere binarie ottenute tramite la binarizzazione dell'immagine di profondità sono praticamente indistinguibili.

4.4.2. Maschere di Profondità ed Edges

Per migliorare la capacità discriminatoria della rete neurale in casi come quello della figura precedente, è necessario aggiungere informazioni all'input. Una prima idea è quella di estrarre gli edges dall'immagine in scala di grigio, utilizzando, per esempio, l'operatore di Sobel. Il risultato di questa operazione può essere quindi binarizzato mediante una sogliatura, e dall'immagine così ottenuta si possono calcolare gli istogrammi orizzontale e verticale. Essi, infine, vengono utilizzati insieme agli input precedentemente discussi (maschere di profondità) per generare un vettore 4L-dimensionale (400 elementi per L=100), che potrà essere fornito in input alla rete neurale.

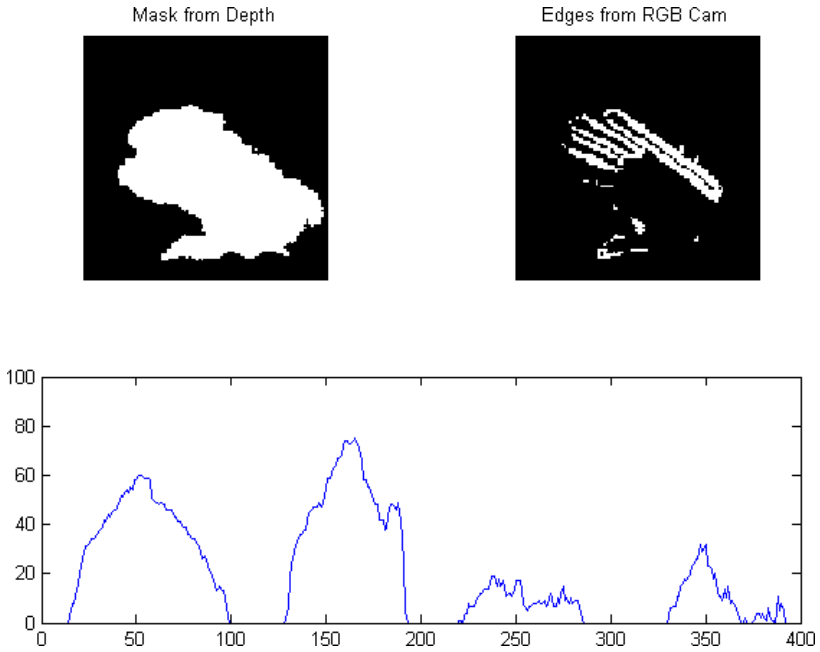


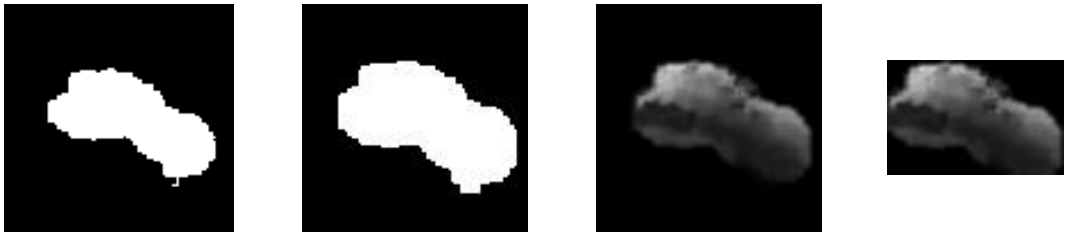
Figura 21. Gli istogrammi di proiezione verticale ed orizzontale della maschera binaria e degli edge, che insieme forniscono il vettore di ingresso alla rete neurale.

Il difetto di questo approccio risiede nella forte dipendenza degli edge dalla qualità dell'immagine, che in genere è sempre meno precisa al crescere della distanza dell'utente dal sensore. Inoltre, in questo modo si perde la possibilità di utilizzare guanti durante il riconoscimento, e l'utilizzo di anelli può ridurre la precisione.

4.4.3. Descrittori SURF

Anzichè utilizzare gli edges come informazione aggiuntiva in grado di discriminare maschere binarie simili, si possono combinare i risultati della binarizzazione dell'immagine di profondità, con le informazioni ottenute dalla videocamera RGB. Sia Im_{mask}* l'immagine ottenuta sottoponendo Im_{mask} ad una operazione morfologica di dilatazione (che ingrandisce la maschera di circa 2 pixel).

Si supponga ora di moltiplicare punto a punto Im_{mask}* per Im_{Gray}[5], e di considerare il minimo rettangolo includente la zona non nera del risultato. Sia Im_{ROI} quest'ultima regione.



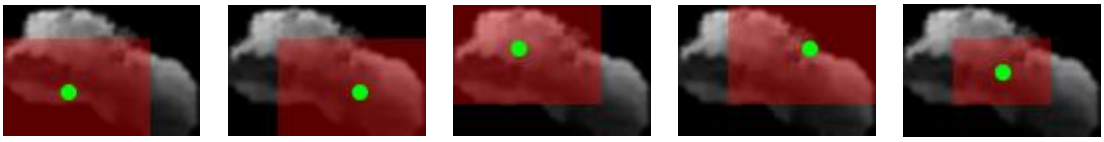
(a) (b) (c) (d)

Figura 22. L'immagine binaria (a) è dilatata morfologicamente (b), ed il risultato è applicato all'immagine in scala di grigi. In questo modo si ottiene l'eliminazione del background (c). A questo punto, si può considerare la sola regione di interesse (Im_{ROI}) (d), rimuovendo le zone irrilevanti di sfondo nero.

SURF (Speed-Up Robust Features) è un rilevatore di caratteristiche locali definito da Bay et al. [29] [28], che viene generalmente utilizzato in computer vision per molti scopi. Esso si basa sul riconoscimento di punti caratteristici (keypoints), ad ognuno dei quali vengono associate dei descrittori che possono essere utilizzati, ad esempio, come termine di confronto per il riconoscimento di oggetti. I descrittori estratti hanno proprietà di invarianza alla rotazione ed alla scala, e l'algoritmo di estrazione è molto veloce, prestandosi molto bene ad implementazioni real time.

[5] Im_{mask}* è supposta a valori reali in [0, 1], perciò la moltiplicazione punto a punto equivale all'applicazione della maschera Im_{mask}* ad Im_{mask}.

Bagdanov et al. [15] utilizzano un metodo che sfrutta i descrittori SURF, con un'approccio diverso da quello tradizionalmente utilizzato. Essi mostrano che è possibile addestrare una SVM utilizzando come ingressi quelli generati dall'estrazione di cinque descrittori SURF-128 (una versione di SURF che produce vettori di descrittori di dimensione 128), calcolati sulla base di altrettanti keypoints. La differenza rispetto all'approccio tradizionale è che i keypoints utilizzati non vengono ottenuti automaticamente mediante SURF, ma vengono definiti esplicitamente prima del calcolo dei descrittori. Ogni keypoint è riferito a porzioni dell'immagine parzialmente sovrapposte, in modo che l'unione delle aree interessate ricopra certamente l'intera immagine. La figura seguente illustra meglio questa suddivisione:



(a) (b) (c) (d) (e)

Figura 23. Scelta dei keypoints e delle porzioni di immagine a cui essi si riferiscono.

Una volta definiti i punti chiave, viene utilizzato l'algoritmo di estrazione per convertire le regioni di interesse in vettori di 128 elementi. In questo modo si estrae dall'immagine un vettore di 128 × 5 = 640 elementi, che possono essere utilizzati come ingresso della rete neurale. Questa scelta consente di sfruttare le proprietà di invarianza alla scala e alla rotazione. Anche in questo caso, tuttavia, si deve assumere l'ipotesi che l'utente non indossi guanti, e che le condizioni di illuminamento siano relativamente accettabili.

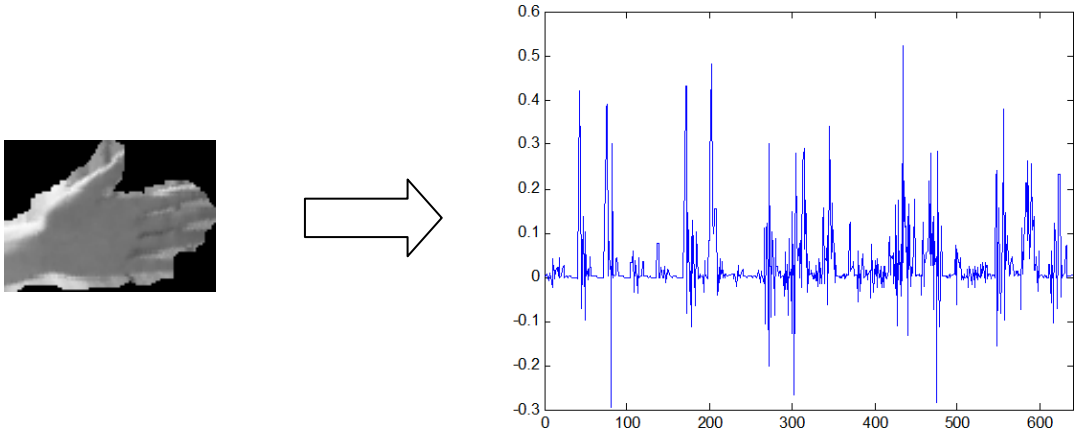


Figura 24. Un esempio che mostra la trasformazione di un'immagine della mano nel vettore SURF utilizzato come input della rete neurale

4.5. Struttura ed Uso della Rete Neurale

Poichè lo scopo del metodo qui descritto è la classificazione dello stato della mano per stabilire se essa è chiusa o meno, il processo decisionale può essere implementato con una rete neurale, come spesso avviene in situazioni di questo tipo [30]. L'addestramento della rete viene effettuato con l'ausilio dell'apposito toolbox di MATLAB, che implementa una variante del metodo di apprendimento con retropropagazione di Widrow-Hoff [33]. La rete ha una struttura costituita da N_I neuroni direttamente connessi agli input, un solo livello nascosto costituito da N_H neuroni, e $N_S = 2$ neuroni di uscita. Essa può, quindi, essere schematizzata come segue:

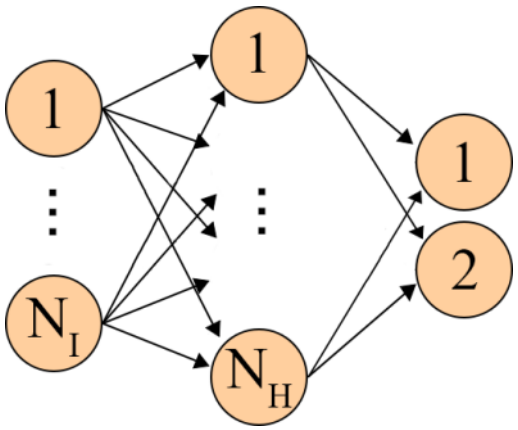


Figura 25. Struttura della rete neurale utilizzata come classificatore

Ogni neurone si avvale di una funzione di trasferimento non lineare per il calcolo dell'uscita, che è la tangente iperbolica. La formula utilizzata da MATLAB, più efficiente dell'implementazione di tanh(n), è la seguente [34]:

$$tansig(n) = \frac{2}{1 + e^{-2n}} - 1$$

Per stabilire il numero di neuroni nascosti N_H, piuttosto che utilizzare una regola del pollice o procedere per tentativi, si sceglie un numero di neuroni che, secondo quanto dimostrato da Elisseeff e Paugam-Moisy [35], si avvicina a quello che consente l'apprendimento esatto. Con la funzione di trasferimento sopra citata, e detti N_P il numero di elementi dell'insieme di addestramento, N_S il numero di uscite della rete e N_I il numero di ingressi, assumendo inoltre nullo il grado di ridondanza

dell'insieme di addestramento, si dimostra che, per consentire alla rete di apprendere esattamente l'insieme di addestramento, deve aversi che:

$$\frac{N_P N_S}{N_I + N_S} \leq N_H \leq 2\frac{N_P N_S}{N_I + N_S}$$

Ciò, ovviamente, non vuol dire che con questo numero di neuroni si giunge all'apprendimento esatto, bensì che è più probabile avvicinarsi a questo apprendimento. Un processo di addestramento accurato, in genere, non consente mai l'apprendimento esatto, per evitare problemi di overfitting.

Scegliamo, quindi, $N_H = 1{,}5 \left\lfloor \frac{N_P N_S}{N_I + N_S} \right\rfloor$.

4.6. Filtraggio dell'Uscita

Poichè l'uscita della rete neurale è generalmente rumorosa, è bene utilizzare un meccanismo che attenui le imprecisioni dovute al rumore. Per farlo, Bagdanov et al. [15] modellano il rumore dell'uscita come un processo gaussiano a media nulla, e utilizzano un filtro di Kalman per attenuarne l'effetto. Il presente metodo, invece, utilizza una EWMA (exponentially weighted moving average) temporale applicata sull'uscita. Dato che, in genere, la variazione dell'uscita tra istanti successivi è minima (se lo stato della mano rimane lo stesso), effettuare la media pesata tra l'uscita corrente della rete neurale ($out_{NeuralNetwork}$) e quella precedente (out_{i-1}), consente di attenuare effetti di rumore indesiderati che si verificano saltuariamente, mentre quando c'è una reale transazione di stato, l'effetto di questo meccanismo è un leggero ritardo nella risposta, che può comunque essere tollerato:

$$out_i = (1 - \alpha) \times out_{i-1} + \alpha \times out_{NeuralNetwork}$$

Si vede sperimentalmente che con $\alpha = 0{,}3$ si ottiene un buon compromesso tra la rimozione del rumore e la reattività del sistema.

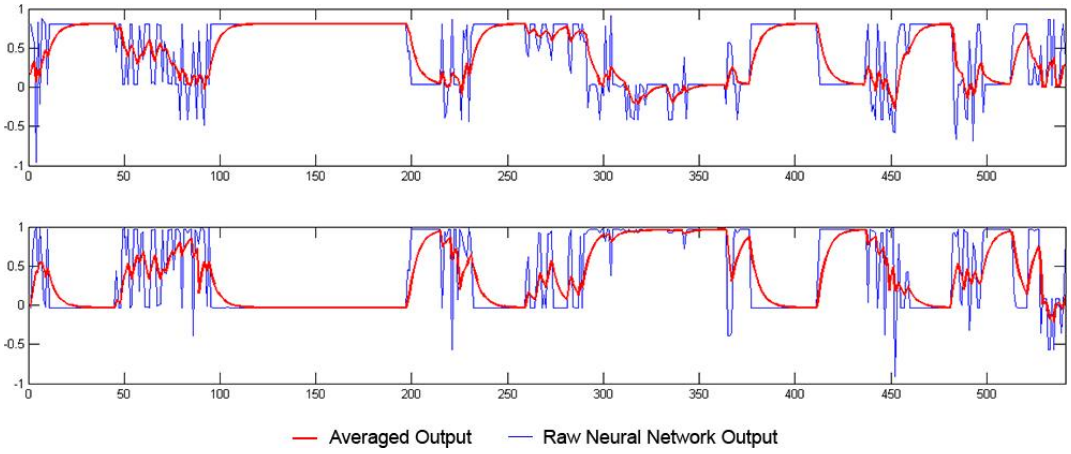


Figura 26. Confronto tra l'uscita della rete neurale senza media temporale, e l'uscita mediata

5. Risultati Sperimentali

5.1. Confronto tra i metodi di elaborazione dell'input

La verifica della bontà del metodo descritto è stata condotta analizzando le uscite di diverse reti neurali, ognuna delle quali è stata costruita in accordo con ciasuna delle tre possibili rappresentazioni della mano come input della rete, sopra descritte.

- Per il metodo basato sull'utilizzo delle sole maschere binarie di profondità, si è addestrata una rete neurale con un insieme di addestramento di circa 2500 esempi, con un livello nascosto costituito da $N_H = 36$ neuroni.
- Per il metodo basato sull'utilizzo sia delle maschere binarie di profondità, che degli edges, si è addestrata una rete neurale con un insieme di addestramento di circa 2500 esempi, con un livello nascosto costituito da $N_H = 20$ neuroni.
- Per il metodo basato sull'utilizzo dei descrittori SURF, si è addestrata una rete neurale con lo stesso insieme di addestramento utilizzato da Bagdanov et al. [15], di circa 28400 esempi, con un livello nascosto costituito da $N_H = 133$ neuroni.

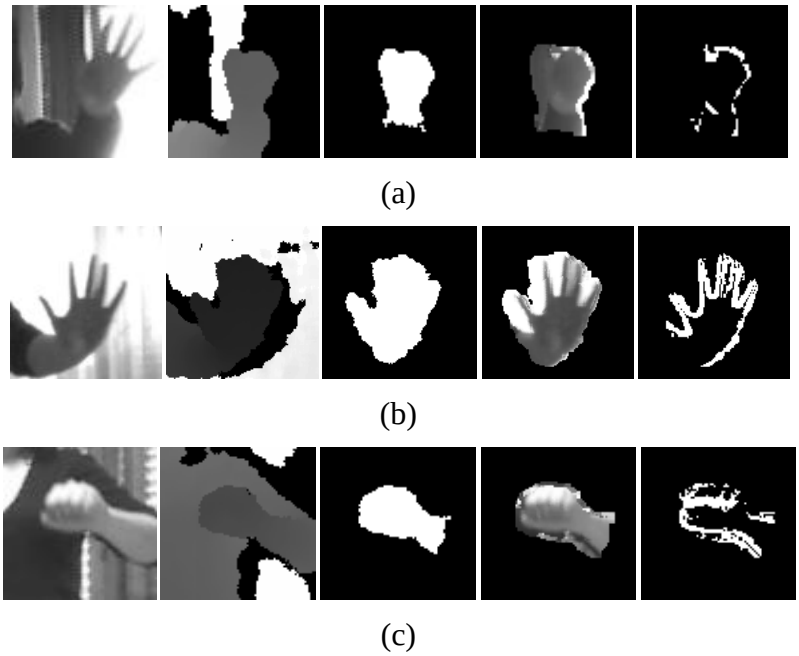
Il numero dei neuroni nascosti N_H è stato calcolato secondo quanto esposto precedentemente.

La tabella seguente mostra i tempi richiesti per l'elaborazione, su un MacBook Pro 15" Late 2011, dotato di processore quadcore a 2,4 GHz [36]. L'approccio più semplice tra quelli visti (e cioè quello basato sulle sole maschere binarie di profondità) è risultato essere anche il più veloce. In ogni caso, però, tutti e tre i metodi si prestano per applicazioni real-time, in quanto consentono un ipotetico funzionamento fino a 100 fps; poichè il sensore Kinect fornisce dati con una frequenza massima di 30 fps, questo risultato è più che soddisfacente.

Depth Mask Only	**Depth Mask & Edges**	**SURF Features**
1,6865 ms	10,1584 ms	7,6415 ms
$3{,}8888 \cdot 10^3$ ticks	$31{,}4383 \cdot 10^3$ ticks	$13{,}2570 \cdot 10^3$ ticks

Figura 27. Confronto tra i tempi di elaborazione del sistema

I test sono stati effettuati con utenti aventi sia le maniche della maglietta alzate, sia abbassate, in 3 tipologie di ambiente. In ognuna di esse, il background non era mai uniforme, ed erano presenti sullo sfondo oggetti di vario genere, e con diverse caratteristiche riflettenti. Un primo ambiente era caratterizzato da un'illuminazione pressocchè costante; un secondo da una illuminazione mediamente uniforme, ma meno intensa (rendendo inferiore il contrasto dell'intera scena); l'ultimo ambiente (più critico) era invece caratterizzato da una grande sorgente luminosa posta dietro l'utente (una finestra aperta). Il sensore Kinect è stato attivato utilizzando la funzionalità di regolazione automatica dell'esposizione.



(a)

(b)

(c)

Figura 28. Alcuni esempi utilizzati per i test: (a) illuminazione intensa retrostante l'utente, che "confonde" il sensore, impedendo il riconoscimento (mano aperta confusa come mano chiusa); (b) mano aperta riconosciuta correttamente nello stesso ambiente; (c) mano chiusa correttamente riconosciuta nell'ambiente caratterizzato da illuminazione uniforme

I dati seguenti mostrano gli errori di classificazione in termini percentuali, al variare della distanza e delle condizioni di illuminamento dell'ambiente.

Hand Depth Mask Only as Neural Network Input

		ENVIRONMENTS						Average
		Constant illumination		Medium illumination		Rear illumination (opened window)		
		Sleeves	No sleeves	Sleeves	No sleeves	Sleeves	No sleeves	
DISTANCE	1.5 m	3.47%	3.83%	9.55%	3.93%	18.59%	3.78%	7.19%
	2.0 m	8.09%	14.72%	9.54%	2.03%	23.51%	13.45%	11.89%
	2.5 m	10.46%	6.51%	15.65%	15.04%	36.50%	31.11%	19.21%
Average		7.84%		9.29%		21.16%		**12.76%**

Distance ↓

Lighting quality ←

Hand Depth Mask and Edges as Neural Network Input

		ENVIRONMENTS						Average
		Constant illumination		Medium illumination		Rear illumination (opened window)		
		Sleeves	No sleeves	Sleeves	No sleeves	Sleeves	No sleeves	
DISTANCE	1.5 m	10.40%	13.79%	32.16%	7.25%	33.96%	21.41%	19.83%
	2.0 m	21.32%	25.08%	17.31%	22.72%	26.32%	19.70%	22.08%
	2.5 m	41.61%	28.77%	30.22%	28.52%	36.06%	37.11%	33.71%
Average		23.49%		23.03%		29.09%		**25.21%**

Distance ↓

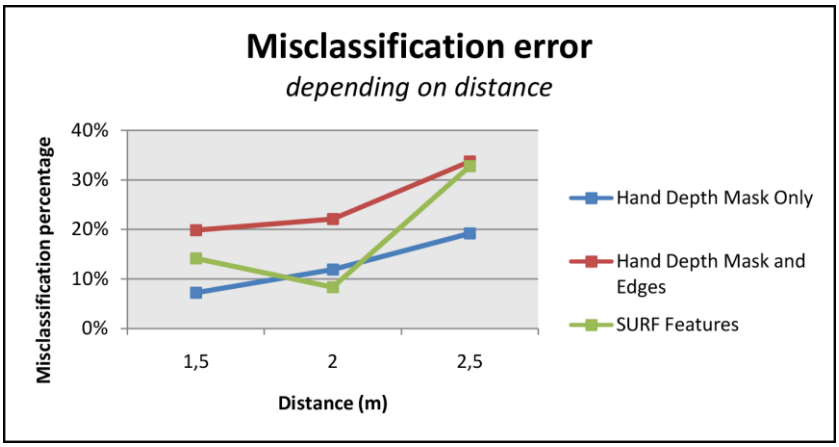
Lighting quality ←

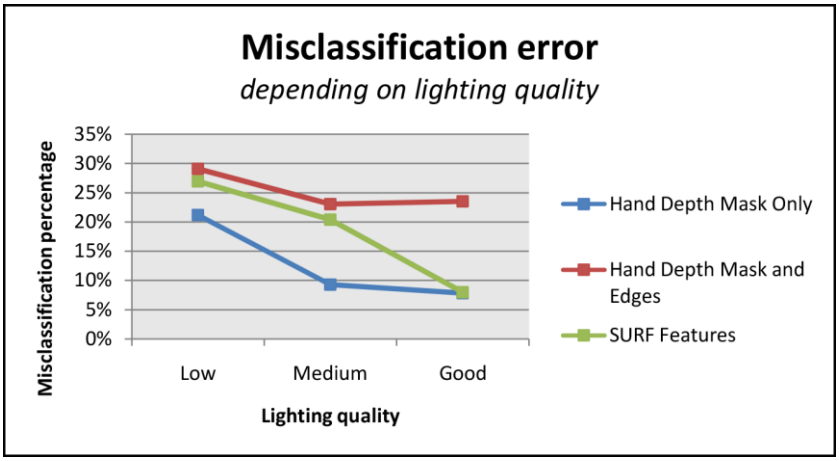
Hand SURF Features as Neural Network Input

		ENVIRONMENTS						Average
		Constant illumination		Medium illumination		Rear illumination (opened window)		
		Sleeves	No sleeves	Sleeves	No sleeves	Sleeves	No sleeves	
DISTANCE	1.5 m	4.46%	4.98%	26.13%	12.99%	17.84%	18.64%	14.17%
	2.0 m	1.23%	0.67%	6.71%	8.32%	15.79%	17.42%	8.36%
	2.5 m	28.95%	7.53%	43.70%	24.41%	52.21%	39.78%	32.76%
Average		7.97%		20.38%		26.95%		**18.43%**

Distance ↓

Lighting quality ←





Da quanto sopra riportato, si evince che il metodo basato sulle maschere di profondità, oltre ad essere il più semplice e veloce, risulta anche quello mediamente migliore. Ciò si spiega con la sostanziale indipendenza della risposta del sistema dalle variazioni di luminosità. Queste ultime, invece, sono proprio quelle che rendono poco efficiente il metodo basato sulla valutazione degli edge. L'algoritmo di Sobel, infatti, è molto sensibile alle variazioni di luminosità, che lo rendono poco adatto ad ambienti non controllati. Inoltre, al crescere della distanza, le informazioni sugli edge della mano diventano troppo difficili da intercettare. Per quanto riguarda, infine, il metodo basato sull'estrazione dei descrittori SURF, esso risulta essere il migliore in situazioni in cui l'illuminazione è pressocchè costante o con variazioni poco significative, ma risulta inadatto in ambienti con illuminazione poco controllata.

5.2. Confronto con altri metodi della letteratura

Confrontando il metodo descritto con le altre possibilità fornite dalla comunità scientifica, le principali differenze riguardano tre aspetti: vincoli, performance e accuratezza. Si prenderà come riferimento l'opzione basata sull'utilizzo delle maschere di profondità come input del sistema di riconoscimento, dal momento che tale scelta produce i risultati migliori sia da un punto di vista dell'accuratezza che in termini di prestazioni.

5.2.1. Vincoli

Alcuni autori impongono agli utenti restrizioni più o meno forti circa l'ambiente, l'illuminazione o la possibilità di indossare guanti o gioielli. Il sistema descritto in questo articolo è praticamente indipendente dall'illuminazione, in quanto si basa sull'utilizzo delle sole informazioni di profondità. Il sistema di riconoscimento studiato da Bagdanov et al. [15], che si basa sull'utilizzo dei descrittori SURF, necessita di un'illuminazione minima per consentire il riconoscimento della mano (dato che vengono utilizzate le informazioni di colore). Lo stesso vale per tutti gli studi che basano la segmentazione sulle informazioni di colore [21] [20]. Sfruttando le sole informazioni di profondità, infatti, non c'è alcun vincolo significativo sul colore della pelle, nè sulla possibilità di indossare guanti, anelli o bracciali (cosa che è necessaria, ad esempio, nella proposta di Ren et al. [22], in cui l'utente deve indossare un braccialetto per identificare la linea di separazione tra la mano e l'avambraccio). Ciò implica che si potrebbe ipotizzare l'applicazione del metodo basato sulle sole informazioni di profondità anche in quelle situazioni in cui le mani potrebbero essere dipinte (si pensi all'ambito ludico, in cui alcuni bambini possano interagire con la Kinect e contemporaneamente utilizzare pennelli e colori). Inoltre, alcune malattie dermatologiche (come la vitiligine) comportano la comparsa di chiazze chiare sulla pelle, che non hanno rilevanza sulle mappe di profondità, mentre possono causare problemi nel riconoscimento basato sul colore della pelle.

L'unica limitazione significativa di questo metodo è quella relativa alla distanza dell'utente dal sensore, che dovrebbe essere compresa tra gli 1,5 e i 2,5

metri. Bagdanov et al. [15] dichiarano che il loro sistema è funzionante tra 1 e 3 metri, ottenendo talvolta dei buoni risultati anche a distanze maggiori.

5.2.2. Performance

Il sistema descritto effettua il riconoscimento in real time (dal momento che può lavorare a più di 30 fps, frequenza di acquisizione standard di Microsoft Kinect). Questo requisito viene garantito anche da Bagdanov et al. [15], che tuttavia necessita dell'estrazione dei descrittori SURF, riducendo la capacità di elaborazione a 20 fps su un processore single core a 2,8 GHz. Per quanto quest'ultima operazione sia efficiente, è certamente meno veloce dell'utilizzo delle maschere di profondità, che vengono calcolate mediante una semplice operazione di sogliatura. Inoltre, l'elaborazione avviene su ogni frame utile. Il sistema di Ahmed [26], che estrae un gruppo di caratteristiche dall'immagine sogliata e le utilizza per la classificazione, è un esempio in cui l'elaborazione per il riconoscimento viene effettuata soltanto se il frame corrente differisce in modo "rilevante" dall'ultimo che è stato elaborato. Ciò impedisce al sistema di considerare tutti i frame ottenuti, ma si rende necessario per ragioni di reattività. Nel sistema qui descritto, la capacità di rispondere in tempo reale viene garantita anche con l'elaborazione frame-per-frame.

5.2.3. Accuratezza

L'unico sistema attenzionato che si occupa del riconoscimento delle sole due pose "mano aperta" e "mano chiusa" è quello di Bagdanov et al. [15], mentre altri [26] [22] [20] consentono il riconoscimento di più pose della mano, a patto che la distanza dal sensore sia sufficientemente limitata. Bagdanov et al. affermano che il loro sistema raggiunge il 98% di accuratezza, risultando di poco migliore rispetto a quello descritto nel presente elaborato (a spese, come già detto, di un onere computazionale leggermente maggiore, e di una meno generale possibilità di applicazione).

6. Applicazioni

Il metodo descritto può essere utilizzato in una vasta gamma di applicazioni che si basano sull'interazione tra uomo e macchina. In particolare, tutti quegli ambiti in cui sono necessarie operazioni di drag and drop, puntamento e selezione di oggetti possono usufruire del riconoscimento dello stato delle mani per consentire l'interazione con l'utente.

In questo paragrafo, si presenteranno alcune applicazioni di esempio (una delle quali è stata effettivamente implementata), allo scopo di evidenziare le possibilità offerte dal sistema descritto nel presente elaborato.

6.1. Implementazione di una Photo Gallery Interattiva

Le principali potenzialità che offre Microsoft Kinect, dovute alla capacità di riconoscere lo scheletro del corpo, risiedono nella possibilità di utilizzare i gesti per eseguire azioni. Ad esempio, un movimento del braccio destro, che si sposta da destra verso sinistra, può essere interpretato come la volontà dell'utente di scorrere una lista orizzontale verso sinistra. Combinare più gesti di questo tipo (eventualmente utilizzando entrambe le braccia) amplia ulteriormente il campo di applicazione.

Il presente elaborato non si occupa del riconoscimento dei gesti, ma sono state studiate ed implementate diverse soluzioni che consentono di associare a sequenze di gesti, delle specifiche azioni. Una di queste, ad esempio, è Kinect DTW [37], che consente di registrare una sequenza di posizioni dei giunti, associarla ad un particolare gesto ed, in seguito, di rilevarla.

Il processo di riconoscimento dello stato della mano, che la distingue tra chiusa e non, non deve avere lo scopo di rimpiazzare le applicazioni per le quali Kinect è stata pensata; piuttosto, esso deve essere utilizzato come strumento aggiuntivo per estendere le funzionalità già esistenti. Per questo motivo, realizzare una galleria fotografica sottoforma di slideshow non richiederebbe l'utilizzo del sistema qui descritto, poichè una lista di foto potrebbe essere molto più intuitivamente controllata con un gesto del braccio. È tuttavia possibile sfruttare la

capacità di riconoscere lo stato della mano per implementare altre funzioni che possono essere integrate, come la selezione, il ridimensionamento, lo spostamento e la rotazione di una foto.

Per prima cosa, quando viene riconosciuto lo scheletro, è possibile mappare una zona dello spazio su un'area dello schermo, e su quest'ultimo mostrare uno o due cursori rappresentanti ciascuno una delle mani dell'utente. L'immagine seguente mostra un'area del corpo che (sulla base dell'esperienza) consente una intuitiva identificazione della posizione delle mani quando queste vengono proiettate sullo schermo.

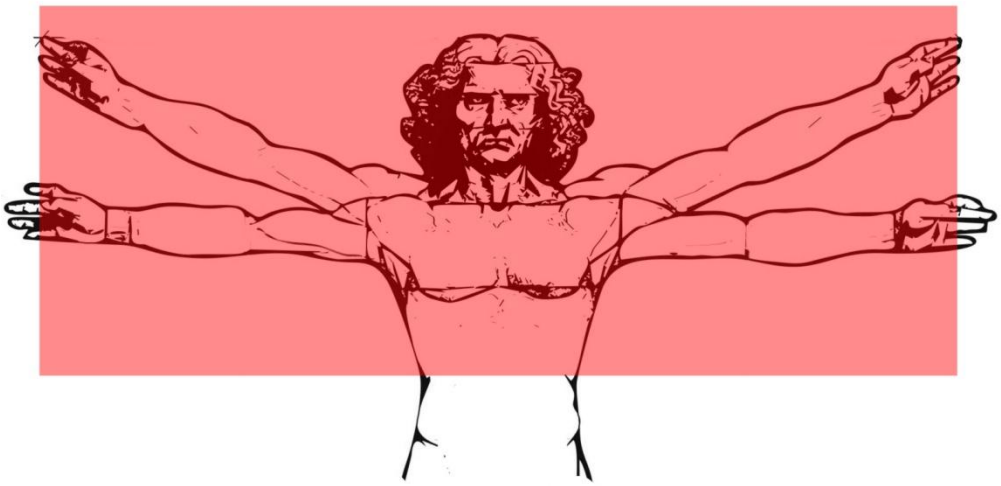


Figura 29. La zona evidenziata in rosso è quella che viene proiettata sullo schermo, consentendo il controllo dei cursori associati ad ognuna delle mani in modo abbastanza intuitivo.

La mappatura dei punti corrispondenti alle mani sullo schermo, senza un'operazione di filtraggio temporale, rende poco agevole il controllo dei cursori, a causa del rumore insito nel funzionamento del sensore. Per questo motivo, in accordo con quanto suggerito da Microsoft per il filtraggio della posizione dei giunti dello scheletro [38], si può utilizzare un filtro di smoothing che si basa su una media mobile temporale della posizione delle mani, definita come segue:

$$\hat{X}(t) = \sum_{i=0}^{N-1} \alpha_i \cdot X(t-i)$$

Qui $\hat{X}(t)$ rappresenta l'uscita filtrata, mentre i valori $X(t-i)$ sono gli N campioni grezzi ottenuti agli istanti precedenti. I pesi α_i sono tali che la loro somma sia pari ad 1. In particolare, nell'implementazione utilizzata, si è scelto N = 6, con i seguenti valori per gli α_i:

$$\alpha_0 = 0{,}30 \quad \alpha_1 = 0{,}20 \quad \alpha_2 = 0{,}15$$
$$\alpha_3 = 0{,}15 \quad \alpha_4 = 0{,}10 \quad \alpha_5 = 0{,}10$$

Una volta filtrata la posizione dei giunti rappresentanti le mani destra e sinistra, si può utilizzare il riconoscitore dello stato della mano per capire se ognuna delle due è aperta o chiusa, associando a tali configurazioni (o a combinazioni di esse, se si utilizzano entrambe le mani) delle azioni opportune. Se, per esempio, sullo schermo vengono visualizzate una o più immagini, la mano chiusa può rappresentare un click, una selezione ed eventualmente un'operazione di drag and drop. Mantenere entrambe le mani chiuse ed all'interno dell'area dell'immagine può servire ad effettuare la scalatura e la rotazione dell'immagine, parametrizzate in funzione della distanza reciproca tra le due mani e dell'angolazione che la congiungente forma rispetto all'orizzontale.

Le immagini seguenti riassumono le modalità di utilizzo dell'applicazione implementata, utilizzabile per la manipolazione di una foto sullo schermo.

a)

b)

c)

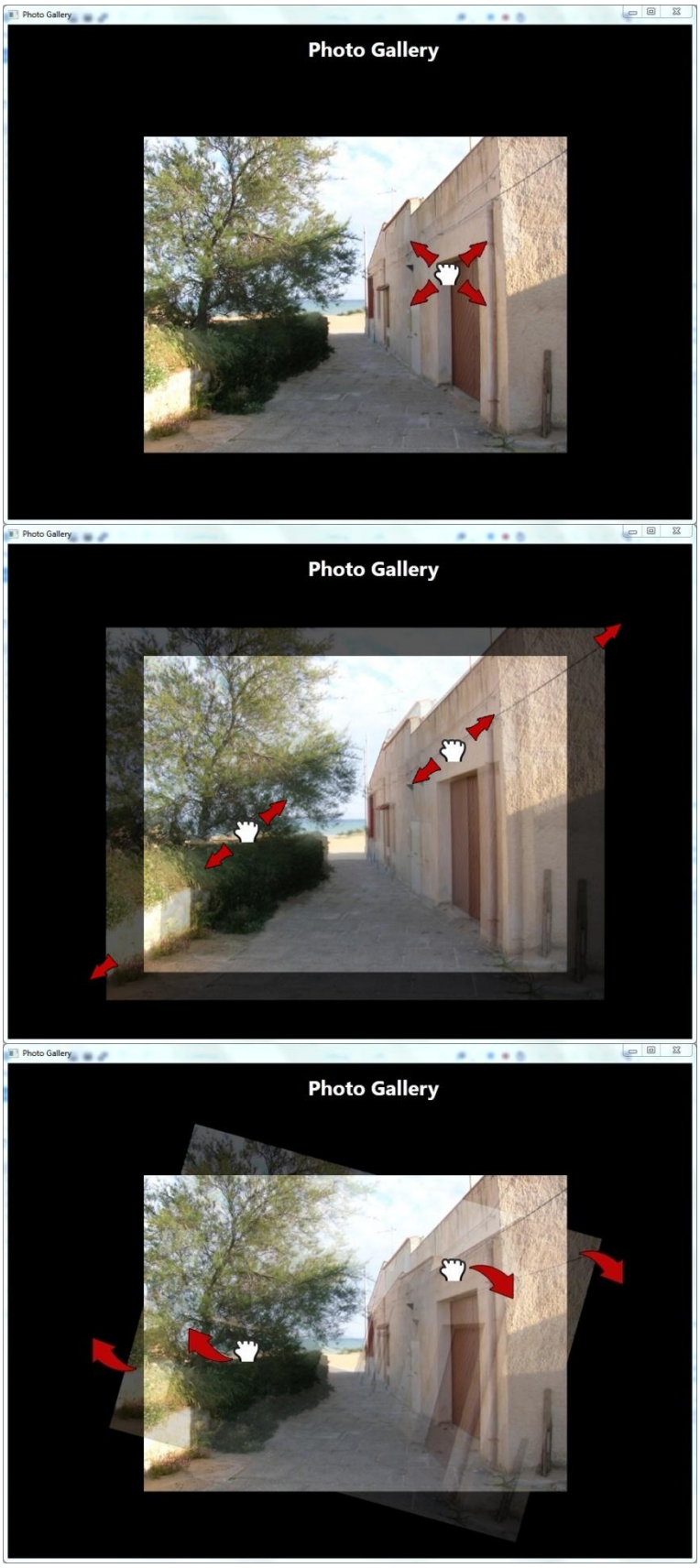


Figura 30. Implementazione delle funzioni di manipolazione di una foto, integrabili in una photo gallery interattiva. Le figure descrivono tre operazioni possibili di manipolazione di immagini: (a) traslazione, (b) scalatura e (c) rotazione.

6.2. Altre possibili applicazioni

6.2.1. Combinazione con videoproiezioni

L'utilizzo di videoproiezioni in combinazione con le funzionalità offerte da Microsoft Kinect consente di ampliarne ulteriormente le possibilità applicative. Piuttosto che utilizzare un normale monitor, le immagini con cui l'utente deve interagire possono essere proiettate su oggetti di vario tipo, per applicazioni che hanno a che vedere con il video mapping, utilizzato in contesti di architettura e design, o per installazioni suggestive di musei e mostre. Proiettando immagini su oggetti alla portata dell'utente, come tavoli di lavoro, o semplicemente utilizzando una parete, si può interagire con esse sfruttando il sistema di riconoscimento qui descritto, associando cioè lo stato della mano ad azioni che possono ripercuotersi sulle immagini proiettate.

6.2.2. Interazione con modelli tridimensionali

Può essere utile estendere le stesse operazioni descritte per il caso della photo gallery interattiva anche ad altri contesti. Un esempio è quello che riguarda la manipolazione di oggetti tridimensionali, per applicazioni di visione virtuale. Dal momento che i giunti ottenuti dallo scheletro contengono informazioni circa la loro posizione tridimensionale nello spazio, il movimento delle mani può essere utilizzato per effettuare trasformazioni di modellazione su poliedri, parametrizzate sulla base del movimento delle mani, della loro distanza e dell'angolo che formano rispetto all'orizzontale. La chiusura e apertura delle mani può essere utilizzata per stabilire quale trasformazione deve essere effettuata, tra rotazione, scalatura o qualsiasi altro tipo di operazione.

6.2.3. Applicazioni ludico-educative

Considerato che Microsoft Kinect è stato inizialmente pensato proprio per applicazioni ludiche, non stupisce il fatto che molte delle applicazioni siano state sviluppate per questo genere di scopi. Oltre all'utilizzo in combinazione con la console Xbox 360, però, è facile immaginare molte altre applicazioni, che uniscono il gioco alla sfera educativa. Di questo particolare ambito si occupa la community di KinectEDucation [39], che si propone come promotore dell'uso di Microsoft Kinect

a fini didattico-educativi, offrendo una vasta gamma di idee ed applicazioni. Tra queste, vale la pena citare:

- applicazioni che estendono il concetto di "gioco delle ombre", tramite il controllo di figure animate proiettate su una parete, sfruttando i giunti dello scheletro;
- applicazioni per l'apprendimento della musica;
- applicazioni per l'apprendimento e la presentazione di concetti di anatomia e scienza, integrando le informazioni ottenute dal sensore con conoscenze circa, per esempio, gli organi del corpo umano o le stelle ed i pianeti;
- applicazioni per facilitare l'apprendimento del linguaggio dei segni;
- e così via...

Ipotizzare l'utilizzo del sistema qui descritto per ampliare le funzionalità di applicazioni di questo tipo è facile. Con le operazioni di puntamento e drag and drop, infatti, è possibile dotare tutti questi sistemi di una larga serie di operazioni secondarie ma necessarie. Per esempio, si potrebbero selezionare oggetti, o semplicamente utilizzare il segnale di mano chiusa come sostituto del click del mouse. Ovviamente, la manipolazione di oggetti, già descritta in precedenza, può essere riportata ed utilizzata in molti ambiti ludico-educativi.

7. Conclusioni

Il metodo descritto si propone come base per sviluppi futuri, che ne migliorino le performances. Mentre è sostanzialmente da escludere l'approccio basato sugli edge (a meno di metodi che rendano indipendente dall'illuminazione il processo di estrazione degli edge), potrebbe essere interessante studiare il perfezionamento degli altri due. Vale inoltre la pena sottolineare che l'utilizzo dei descrittori SURF implica la necessità di utilizzare un algoritmo brevettato, per l'utilizzo del quale è richiesto (per scopi commerciali) il pagamento di royalties. Sfruttando la semplicità e le migliori performance offerte dall'approccio basato sulle maschere di profondità, il metodo qui presentato è esente da ogni tipo di licenza, e può essere applicato a sistemi commerciali senza costi aggiuntivi.

Ad ogni modo, il metodo di riconoscimento basato sull'estrazione dei descrittori SURF potrebbe essere perfezionato migliorando la qualità delle immagini da cui ricavare i descrittori, in modo che questi ultimi dipendano il meno possibile da informazioni di poco interesse. Una prima miglioria potrebbe consistere nell'effettuare uno smoothing della maschera da usare per la rimozione del background, in modo da ridurre la dipendenza dei descrittori dagli edges che separano il bordo della mano dalle porzioni di background nere. Si potrebbero, inoltre, aggiungere altre operazioni di pre-processing, al fine di normalizzare il contrasto e rendere quanto più possibile indipendente dalla luminosità l'immagine.

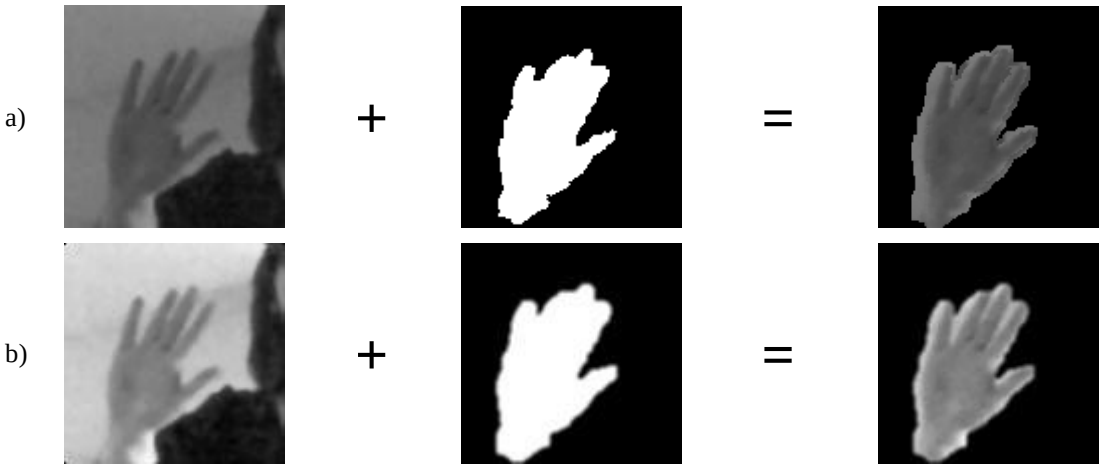


Figura 31. Possibile miglioramento delle immagini da cui estrarre i descrittori SURF. La riga (a) mostra i risultati intermedi dell'algoritmo correntemente utilizzato. Nella riga (b), invece, l'immagine viene prima migliorata con uno stretching di contrasto, mentre la maschera di profondità viene perfezionata con operazioni di filtro mediano, dilatazione e smoothing.

L’approccio basato sull’utilizzo delle maschere di profondità può essere perfezionato modificando la rappresentazione delle immagini. Piuttosto che utilizzare gli istogrammi verticale ed orizzontale, infatti, le maschere di profondità possono essere rappresentate in altri modi, al fine di renderle indipendenti dalla scala, dalla rotazione, eccetera. Una possibilità è quella adoperata da Ren et al. [22], che estraggono una signature dalla maschera di profondità calcolando, al variare dell’angolo, la distanza tra il centroide della mano e il boundary. Il vettore così ottenuto ha una dimensione fissa: se, ad esempio, si campiona ad ogni grado sessagesimale, si ottengono 360 dimensioni, che possono essere ridotte o aumentate modificando il passo di campionamento. Tale vettore può, quindi, essere utilizzato per rappresentare la maschera di profondità, risultando indipendente dalla scala. Per l’indipendenza rotazionale, si potrebbe preventivamente applicare un apposito algoritmo che ruoti opportunamente l’immagine (come Matos et al. [18] suggeriscono). Approcci diversi, come quelli di Ahmed [26] e Biswas et al. [14] utilizzano, invece, un approccio per l’estrazione di 33 caratteristiche da un’immagine binaria, basate sulla percentuale di pixel bianchi in diverse porzioni sovrapposte dell’immagine, nonchè sul calcolo di alcuni momenti centrali della posizione della mano. Questi risultano indipendenti dalla scala, ma potrebbero non rispondere bene a situazioni in cui la risoluzione delle immagini è troppo bassa.

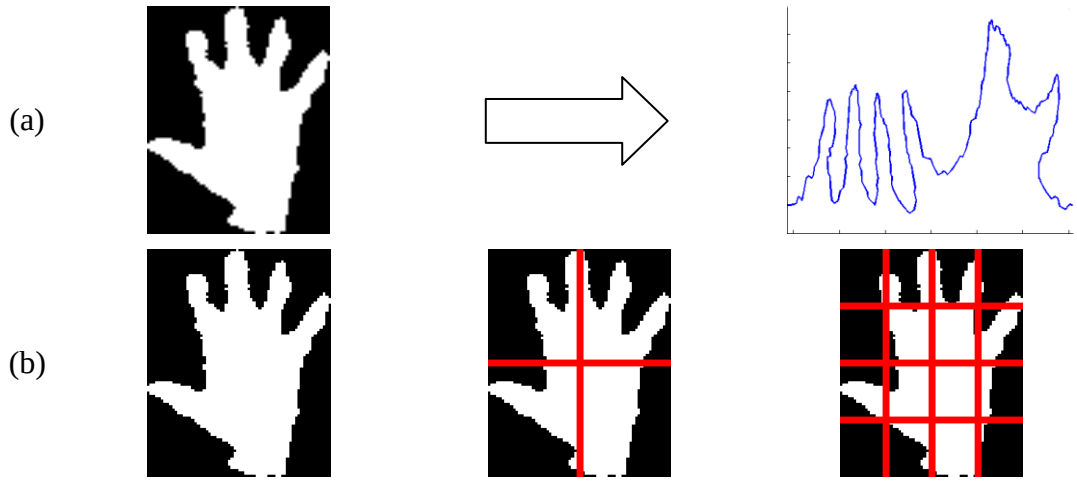


Figura 32. Alcune possibili alternative per la rappresentazione delle maschere binarie di profondità: (a) una signature che rappresenta il contorno come funzione 1D, sfruttando le coordinate polari, utilizzata da Ren et al. [22]; (b) livelli di suddivisione per estrarre alcune caratteristiche proposte da Ahmed [26].

Un’ulteriore modifica potrebbe riguardare il filtraggio dell’uscita. Oltre al filtro di Kalman, utilizzato da diversi autori [15] [16], potrebbe essere semplicemente migliorato lo smoothing dell’uscita sfruttando un filtro di media mobile che considera più campioni temporali precedenti, piuttosto che uno solo, a spese di un leggere incremento del carico computazionale.

Questi ed altri accorgimenti possono essere utilizzati al fine di perfezionare il metodo descritto.

8. Bibliografia e Sitografia

[1] Microsoft. Kinect for Windows - Product Features. [Online]. http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx

[2] Barak Freedman, Alexander Shpunt, Meir Machline e Yoel Arieli, "Depth mapping using projected patterns," US 20100118123, 13 Maggio 2010.

[3] iFixit. Microsoft Kinect Teardown. [Online]. http://www.ifixit.com/Teardown/Microsoft-Kinect-Teardown/4066/

[4] Microsoft. MSDN - Skeletal Tracking. [Online]. http://msdn.microsoft.com/en-us/library/hh973074.aspx

[5] Luigi Maggio. (2012) Il dispositivo Microsoft Kinect. [Online]. http://www.luigimaggio.altervista.org/documenti/tesinaRealtaVirtualeKinect.pdf

[6] Jamie Shotton et al., "Real-time human pose recognition in parts from single depth images," in *2011 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, USA, 2011, pp. 1297-1304.

[7] Jamie Shotton et al. (2011) Real-time human pose recognition in parts from single depth images: Supplementary Material. [Online]. http://research.microsoft.com/pubs/145347/SupplementaryMaterial.pdf

[8] OpenNI. OpenNI The standard framework for 3D sensing. [Online]. http://www.openni.org/

[9] Itseez. OpenCV. [Online]. http://opencv.org/

[10] Emgu CV. [Online]. http://www.emgu.com

[11] Microsoft. MSDN - CoordinateMapper Class. [Online]. http://msdn.microsoft.com/en-us/library/jj663707.aspx

[12] Wonwoo Lee. (2011, Marzo) Kinect Color - Depth Camera Calibration. [Online]. http://cv4mar.blogspot.it/2011/03/kinect-color-detph-camera-calibration.html

[13] Kourosh Khoshelham, "Accuracy analysis of kinect depth data," in *ISPRS workshop laser scanning 2011*, Calgary, Canada, 2011, pp. 133-138.

[14] K. K. Biswas e Kumar Basu Saurav, "Gesture Recognition using Microsoft Kinect®," in *5th International Conference on Automation, Robotics and Applications (ICARA)*, Wellington, New Zealand, 2011, pp. 100-103.

[15] Andrew D. Bagdanov, Alberto Del Bimbo, Lorenzo Seidenari e Lorenzo Usai, "Real-time hand status recognition from RGB-D imagery," in *21st International Conference on Pattern Recognition*, Tsukuba, Japan, 2012, pp. 2456-2459.

[16] Valentino Frati e Domenico Prattichizzo, "Using Kinect for hand tracking and rendering in wearable haptics," in *IEEE World Haptics Conference 2011*, Instanbul, Turkey, 2011, pp. 317-321.

[17] Nobuyuki Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62-66, Gennaio 1979.

[18] Hélder Matos, Hélder P. Oliveira e Filipe Magalhães, "Hand-Geometry Based Recognition System A Non Restricted Acquisition Approach," in *9th International Conference on Image Analysis and Recognition (ICIAR)*, Aveiro, Portugal, 2012, pp. 38-45.

[19] M. K. Bhuyan, Raj Neog Debanga e Kumar Kar Mithun, "Fingertip Detection for Hand Pose Recognition," *Internation Journal on Computer Science and Engineering (IJCSE)*, vol. 4, no. 3, 2012.

[20] Cheng Tang, Yongsheng Ou, Guolai Jiang, Qunqun Xie e Yangsheng Xu, "Hand Tracking and Pose Recognition via Depth and Color Information," in *Proceedings of the 2012 IEEE Internation Conference on Robotics and Biomimetics*, Guangzhou, China, 2012, pp. 1104-1109.

[21] Matthew Tang. (Marzo 2011) Recognizing Hand Gestures with Microsoft’s Kinect. [Online]. http://www.stanford.edu/class/ee368/Project_11/Reports/Tang_Hand_Gesture_Recognition.pdf

[22] Zhou Ren, Junsong Yuan e Zhengyou Zhang, "Robust Hand Gesture Recognition Based on Finger-Earth Mover's Distance with a Commodity Depth Camera," in *Proceedings of the 19th ACM international conference on Multimedia*, New York, NY, USA, 2011, pp. 1093-1096.

[23] Yishen Xu, Jihua Gu, Zhi Tao e Di Wu, "Bare Hand Gesture Recognition with a Single Color Camera," in *CISP '09. 2nd International Congress on Image and Signal Processing*, Tianjin, China, 2009, pp. 1–4.

[24] Stéphane Marchand-Maillet e Yazid M. Sharaiha, *Binary Digital Image Processing: A Discrete Approach*.: Academic Press, 2000.

[25] Heng Du e TszHang To. (Dicembre 2011) Hand Gesture Recognition using Kinect. [Online]. http://iss.bu.edu/data/jkonrad/reports/HDTT11-04buece.pdf

[26] Tasnuva Ahmed, "A Neural Network based Real Time Hand Gesture Recognition System," *Internation Journal of Computer Applications*, vol. 59, no. 4, 2012.

[27] Suryanarayan Poonam, Subramanian Anbumani e Mandalapu Dinesh, "Dynamic Hand Pose Recognition using Depth Data," in *International Conference on Pattern Recognition*, Istanbul, Turkey, 2010, pp. 3105-3108.

[28] Herbert Bay, Tinne Tuytelaars e Luc Van Gool, "SURF: Speeded Up Robust Features," in *9th European Conference on Computer Vision (ECCV)*, Graz, Austria, 2006, pp. 404-417.

[29] Herbert Bay, Andreas Ess, Tinne Tuytelaars e Luc Van Gool, "SURF: Speeded Up Robust Features," *Computer Vision and Image Understanding (CVIU)*, vol. 110, no. 3, pp. 346-359, 2008.

[30] Guoqiang Peter Zhang, "Neural Network for Classification: A Survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 30, no. 4, pp. 451-462, Novembre 2000.

[31] NASA. (Luglio 1995) Man-Systems Integration Standards. [Online]. http://msis.jsc.nasa.gov/sections/section03.htm

[32] DINBelg. (2005) DINBelg 2005. [Online]. http://www.dinbelg.be/anthropometry.htm

[33] Mark Hudson Beale, Martin T. Hagan e Howard B. Demuth. NeuralNetwork Toolbox User's Guide. [Online]. http://www.mathworks.it/help/pdf_doc/nnet/nnet_ug.pdf

[34] The MathWorks, Inc. Hyperbolic tangent sigmoid transfer function. [Online]. http://www.mathworks.com/help/nnet/ref/tansig.html

[35] André Elisseeff e Hélène Paugam-Moisy, "Size of multilayer networks for exact learning: analytic approach," in *Advances in Neural Information Processing Systems 9*, Cambridge, MA, USA, 1997, pp. 162-168.

[36] Apple Inc. (Settembre 2012) MacBook Pro (15-inch, Late 2011) - Technical Specifications. [Online]. http://support.apple.com/kb/sp644

[37] Kinect SDK Dynamic Time Warping (DTW) Gesture Recognition. [Online]. http://kinectdtw.codeplex.com/

[38] Microsoft. MSDN - Skeletal Joint Smoothing White Paper. [Online]. http://msdn.microsoft.com/en-us/library/jj131429.aspx

[39] Johnny Kissko. KinectEDucation. [Online]. http://www.kinecteducation.com/