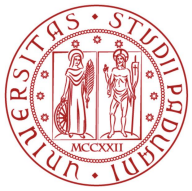


Università degli Studi di Padova

DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

Università degli Studi di Padova

Dipartimento di ingegneria dell'informazione

Corso di Laurea
in Ingegneria Informatica

Confronto sperimentale tra sensori RGB-D e IMU per Human Action Recognition

Relatore:
Prof. Stefano Ghidoni

Correlatore:
Matteo Terreran, PhD

Laureando:
Gianluca Nordio
2007959

Anno Accademico 2022/2023
Data di laurea 19/07/2023

Ai miei nonni,
che mi hanno insegnato la bellezza
della conoscenza e della vita.

Ringraziamenti

Ritengo necessario ringraziare le persone che mi hanno supportato per arrivare compimento di questo lavoro. A partire dal professor Ghidoni per aver fornito, fin da subito, il suo prezioso tempo per ascoltare le idee e fornire consigli, non facendomi mai sentire obbligato nelle scelte. E soprattutto per aver creduto fin da subito nelle capacità degli studenti, più di noi stessi. Un grazie infinito anche a Matteo Terreran, che ci ha instradato sempre nella retta via donando molto tempo e lavoro nei momenti di maggiore difficoltà. Grazie all'IAS-Lab di Unipd e a Giulio per aver fornito i mezzi, i luoghi e un ambiente stimolante.

Un ringraziamento speciale va poi a Giovanni Cinel, compagno di lavoro durante questa tesi e di risate durante gli anni. Senza il supporto che ci siamo dati a vicenda non sarebbe stato possibile portare a termine questo lavoro.

Ai miei famigliari, genitori e a mia sorella per avermi sostenuto per 3 anni minati da periodi difficili per tutto il mondo.

Infine grazie infinito a tutti i compagni di corso come Cristian, Nicola, Moli, Eddie, di vita quotidiana come Paolo, Acc, Pera, Tommy, Antonio e tutta la compagnia.

Sommario

In un mondo dominato dalla crescita in termini di popolarità del Deep learning, ne viene analizzato l'utilizzo per il riconoscimento di azioni (HAR). L'HAR consiste nel predire quale azione è in corso rispetto un insieme predefinito di possibili azioni, fondamentale in diversi contesti come l'intrattenimento e, soprattutto, lo sviluppo dell'industria 4.0. Questa tipologia di industria richiede degli spazi in cui robot e esseri umani possano collaborare in sinergia e affinché ciò sia possibile è cruciale la capacità di riconoscere l'azione che sta svolgendo l'operatore umano. Ad esempio, il robot potrebbe passare all'operatore uno strumento necessaria per l'azione successiva a quella riconosciuta.

L'HAR è possibile a partire da varie tipologie di informazioni come immagini RGB-D, dati grezzi provenienti da sensori inerziali (IMU), oppure la rappresentazione scheletrica (posa del corpo). Lo scheletro può risultare particolarmente vantaggioso rispetto alle altre modalità in quanto rappresentazione leggera e resistente a trasformazioni come rotazioni e traslazioni. Poiché gli scheletri possono essere ottenuti dalle immagini RGB-D o dagli IMU, l'obiettivo di questo lavoro è un confronto, nell'ambito dell'HAR, tra le due tecnologie di produzione. Nello specifico, durante il confronto sperimentale sono stati utilizzati i sensori inerziali Xsens e una rete di telecamere RGB-D Intel.

Per la classificazione sono stati utilizzati dei Multilayer Perceptron (MLP) a cui sono state fornite le posizioni tridimensionali dei giunti del corpo considerate come input. I parametri del modello sono stati selezionati in modo empirico attraverso diversi esperimenti finalizzati a cercare la miglior configurazione.

I modelli finali hanno ottenuto come prestazioni migliori un'Accuracy pari a 91.58% nel caso dei dati ottenuti tramite IMU, mentre un'Accuracy del 81.72% nel caso di dati prodotti dalle immagini RGB-D. Il miglior risultato da parte del riconoscimento tramite IMU è in linea con le ipotesi relative alla maggiore precisione della posa stimata dai sensori inerziali in termini di movimenti relativi del corpo.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

Introduzione

I gesti sono uno dei modi più utilizzati dagli uomini per comunicare tra di loro. Altre metodologie di comunicazione non sempre sono efficaci come, ad esempio, l'utilizzo della voce all'interno di ambienti rumorosi come le celle industriali in cui sono presenti robot in azione. In questi scenari si cerca allora un modo alternativo di comunicare. Motivo per cui nel tempo si è cominciata ad esplorare la possibilità di utilizzare i gesti per poter impartire dei comandi a computer o robot. Ma il compimento di un gesto specifico non può sempre essere possibile, ad esempio alzare una mano mentre si trasporta un oggetto, o può risultare poco naturale rispetto all'azione in corso, ad esempio l'allontanamento di una mano dal luogo in cui si sta lavorando. Diventa quindi necessario poter riconoscere non solo dei gesti predefiniti (ad esempio alzare la mano), ma anche movimenti ed azioni più complesse quali quelle compiute da una persona durante una normale lavorazione, in modo da poterne ricavare informazioni utili. Tale problema prende il nome di riconoscimento delle azioni umane o Human Action Recognition (HAR).

L'impartire comandi ad un computer o robot è solo uno dei possibili casi di applicazione. L'HAR trova applicazione in diversi campi: la sicurezza, la medicina, la riabilitazione, l'intrattenimento e soprattutto la collaborazione uomo-robot. Per quanto concerne la sicurezza l'HAR può essere utilizzata per monitorare l'attività umana in aree sensibili come aeroporti, stazioni ferroviarie, banche, strade trafficate, ecc. in modo da individuare comportamenti sospetti e prevenire situazioni di pericolo. Nella medicina e nella riabilitazione può essere utilizzata per monitorare e valutare la performance fisica degli individui, ad esempio in ambito sportivo o per la riabilitazione post-operatoria. Per l'intrattenimento l'HAR può essere utilizzata per creare esperienze che siano interattive, come videogiochi o simulazioni virtuali, in cui le azioni dell'utente vengono riconosciute e utilizzate

per interagire con l'ambiente virtuale. Infine, il riconoscimento di azioni umane consente di rendere la collaborazione uomo-robot più naturale. Nell'ambito Human Robot Collaboration (HRC) si utilizza l'azione riconosciuta per avere una coesistenza più efficacie e sicura tra le due entità uomo e robot. Ad esempio, un robot che assiste un operatore umano durante l'assemblaggio di un prodotto, può riconoscere quale azione il partner umano sta attualmente svolgendo e passare gli strumenti o pezzi successivi necessari a proseguire l'assemblaggio.

Nello stato dell'arte sono presenti numerose modalità di riconoscimento delle azioni umane (HAR). Si possono infatti distinguere due macrocategorie [1]: le modalità visive e non visive. All'interno delle modalità visive ricadono: immagini RGB, scheletro 3D, immagini di profondità, sequenza di infrarossi, Point cloud e telecamere ad eventi. Mentre nella categoria delle modalità non visive ricadono: dati da sensori inerziali, accelerometri, radar e wifi. Il focus di questo lavoro si pone su una modalità visiva: lo scheletro. Si tratta di una modalità visiva in quanto si tratta di un insieme di punti 3D nello spazio, un punto per ogni giunto del corpo quali spalle, gomiti, ginocchia etc. Ed è facilmente ottenibile, unendo tali giunti 3D con dei segmenti, una rappresentazione visiva della posa del corpo. Saranno confrontate due metodologie differenti per ricavare tale scheletro: partendo da dati prodotti da una rete di videocamere RGB-D, quindi una modalità comunque visiva, e da dati prodotti da sensori inerziali, quindi una modalità non visiva. Nella comparazione vengono valutate le differenze, in termini di prestazioni, nella fase di riconoscimento delle azioni.

Le videocamere RGB-D sono una modalità visiva in quanto forniscono delle immagini RGB, considerate delle rappresentazioni visivamente intuitive. Notando che oltre a tali rappresentazioni viene fornita un'ulteriore informazione: la profondità (da cui il nome RGB-Depth). Mentre i sensori inerziali (IMU - inertial measurement unit) forniscono dati grezzi come accelerazione, velocità angolare e varie altre informazioni. In entrambi i casi queste tipologie di dati vengono convertiti in scheletro utilizzando specifici strumenti: OpenPTrack [2] per le telecamere RGB-D e Xsens MVN Analyze per i sensori inerziali.

Al fine di effettuare il confronto tra le due tipologie di scheletro per il riconoscimento di azioni, è stato creato un dataset contente principalmente azioni affini alla collaborazione uomo-robot. I dati che compongono la collezione sono sequenze RGB, sequenze RBG-D e posizioni dei giunti di scheletri ottenuti da immagini RGB-D e stima della posa del corpo fatta dal software Xsens sulla base dei dati grezzi delle IMU. E' stato poi creato un modello di riconoscimento con

un meccanismo di pre-processing degli scheletri ottenuti dalle due fonti, in modo da verificare le differenti prestazioni in fase di classificazione.

1.1 Human Robot Collaboration

Allo stato attuale l'automazione dei reparti produttivi all'interno delle industrie domina il panorama globale. I robot sono, infatti, in grado di raggiungere una precisione e una velocità inimmaginabili per l'essere umano. Il tutto a bassi costi. In questi ambienti può essere necessaria, in determinate situazioni, la presenza di personale umano. Tuttavia, con la presenza di lavoratori all'interno della zona di lavoro la capacità produttiva dei robot decresce per motivi di sicurezza. A titolo di esempio, si può verificare una riduzione della produttività a causa della necessità per i robot di operare a velocità ridotta in presenza di persone, oppure si possono adottare gabbie di protezione che ostacolano il trasferimento di materiale. Nell'emergente contesto dell'industria 4.0, il concetto di collaborazione ha portato alla necessità di rimuovere la separazione tra ambiente di lavoro umano e ambiente di lavoro dei robot, permettendo una collaborazione efficacie tra le due entità. La rimozione di questa barriera porta con sé dei nuovi rischi e modifiche agli standard di sicurezza [3].

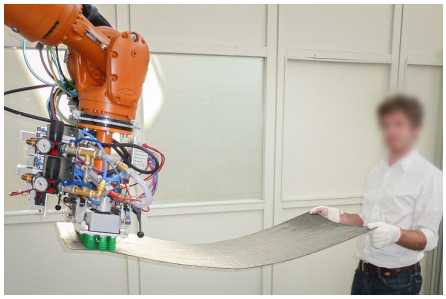


Figura 1.1: Trasporto di fibra di carbonio con ausilio di un robot [4]

Un esempio di questo contesto è il progetto DrapeBot [5], che si occupa del draping di fibra di carbonio, un processo durante il quale strati di fibra di carbonio vengono posizionati su degli stampi. Lo scopo del progetto è la collaborazione uomo-robot, in cui si ha l'assistenza da parte di un robot industriale nel trasporto di grandi pezzi di carbonio (vedi Figura 1.1) e della loro stesura su superfici con curvature leggere. Mentre l'uomo si occupa della copertura nelle zone più difficili.

Affinché operazioni come quella appena citata siano possibili è necessario lo sviluppo di strumenti che consentano una collaborazione ottimale. Nello specifico

un sistema di HRC consiste di quattro elementi basilari [6]: percezione, riconoscimento, predizione e azione. Prima di tutto i sensori quali telecamere, che monitorano l'ambiente di lavoro, producono dei dati per l'analisi delle azioni umane. Successivamente le azioni umane sono riconosciute dai dati ricevuti. Analizzando poi le informazioni fornite dal riconoscimento si ha una predizione dell'azione futura, sulla base della quale il robot può fornire assistenza al lavoratore umano in modo attivo. Questa sequenza spiega il motivo per cui il riconoscimento delle azioni umani ha un ruolo cruciale nell'ambito della collaborazione uomo-robot.

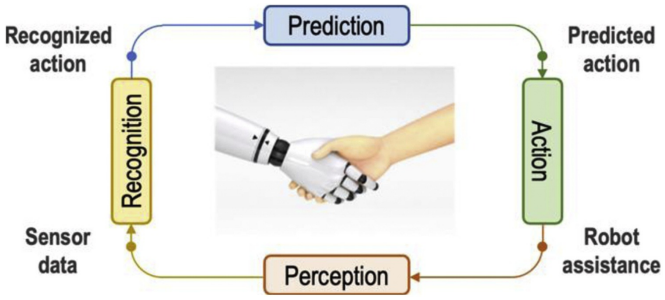


Figura 1.2: Flowchart del processo di HRC [6].

1.1.1 Cobot

Nell'ambito della collaborazione uomo-macchina, oltre a fornire i mezzi per comunicare tra i due, è importante che la macchina riesca a percepire un eventuale urto con il lavoratore. Da questa richiesta nascono i Cobot, termine derivato da robot collaborativi, capaci di lavorare garantendo la sicurezza degli operatori umani. Ciò viene ottenuto attraverso dei sensori posizionati lungo tutto il dispositivo, atti allo scopo di percepire contatti indesiderati, che facciano entrare il Cobot in una modalità sicura. Si tratta di un atteggiamento non noto ai robot tradizionali, che continuerebbero l'azione in corso provocando lesioni a chi si trovi nel suo percorso.

Alcuni dei produttori più famosi di cobot sono Franka Emika [7] e Universal Robots [8]. Si tratta comunque di un mercato in continua espansione, vista la grande versatilità di questi robot. Nonostante i Cobot presentino dei sensori per rilevare l'urto con una persona, in un contesto di utilizzo reale, l'obiettivo è anticipare tali situazioni al fine di evitarle. In caso contrario, sarebbe necessario fermare il robot e interrompere la produzione. Pertanto, è fondamentale riconoscere le azioni delle persone per poter determinare dei movimenti del robot, affinché siano in accordo con ciò che stanno facendo i soggetti.

(a) **(b)**

Figura 1.3: In figura (a) un cobot, mentre in figura (b) un robot industriale.

1.2 RGB-D camera

Una videocamera RGB-D è una videocamera RGB, ovvero a colori, a cui viene aggiunto un ulteriore sensore in grado di misurare la profondità.

La misurazione della profondità può avvenire principalmente in quattro modi: ToF (Time of Flight), luce strutturata, LiDAR (Light Detection and Ranging) e SCS (Stereo Camera Sensing) [9] [10].

Nel caso di ToF si illumina la scena con una sorgente di luce infrarossa e si rileva quella riflessa. La telecamera misura il tempo impiegato dalla luce per viaggiare dalla telecamera alla scena e tornare indietro, e utilizza questa informazione per calcolare la distanza tra la telecamera e l'oggetto. Questo processo viene effettuato per ogni punto del campo visivo. Si tratta, quindi, di un approccio analogo a quello utilizzato da sistemi come radar e sonar.

Mentre i sensori a luce strutturata proiettano un pattern noto sulla scena e misurano la distorsione. Tramite dei calcoli matematici di triangolazione riescono così a definirne la distanza. La proiezione può essere prodotta da fonti di luce appartenente al campo visibile o al campo degli infrarossi.

Per quanto riguarda il LiDAR viene utilizzato un laser che ruotando effettua una misurazione della distanza di alcuni punti con un criterio analogo al ToF. Produce una nuvola di punti sparsa, ovvero, va a mappare solo alcuni dei punti della scena.

SCS è una metodologia molto simile all'utilizzo della luce strutturata in cui però non viene utilizzata la proiezione di luce. Si effettua una triangolazione delle immagini ricevute da più videocamere per estrarre la posizione.

1.3 IMU - Inertial Measurement Unit

I sensori inerziali (IMU) sono dispositivi elettronici utilizzati per definire posizione ed orientamento di un oggetto. Per ottenere questi dati vengono misurati parametri come l'accelerazione, forza applicata, velocità angolare e, in alcuni casi, il campo magnetico. Nello specifico i giroscopi nei sensori inerziali misurano la velocità angolare dell'oggetto, consentendo di determinare l'orientamento e i cambiamenti di rotazione. Gli accelerometri misurano l'accelerazione lineare dell'oggetto, che può essere utilizzata per calcolare la velocità e la posizione. I magnetometri, se presenti, misurano il campo magnetico circostante, che può essere utilizzato per rilevare la direzione del nord magnetico o per stabilire l'orientamento rispetto a un campo magnetico esterno. La combinazione di questi dati provenienti dai sensori inerziali consente di ottenere una stima accurata della posizione e dell'orientamento di un oggetto nello spazio tridimensionale.

Nella vita quotidiana molti utilizzano questi sensori: a partire dai cellulari e arrivando agli smartwatch. Infatti, l'evoluzione dei sensori inerziali ha portato a dispositivi sempre più piccoli, leggeri e precisi. Alcuni sensori inerziali sono integrati in dispositivi indossabili, come smartwatch, braccialetti fitness o visori per la realtà virtuale, consentendo il monitoraggio dei movimenti dell'utente in modo discreto e non invasivo arrivando a riconoscere attività come il tipo di sport che si sta praticando. In commercio esistono, inoltre, set di sensori che, tramite il posizionamento su determinate parti del corpo, riescono a ricostruire uno scheletro attraverso un modello biomeccanico del corpo umano. Uno dei più grandi produttori di suite inerziali è Xsens, con i numerosi set in vendita. Tra cui il kit MVN Awinda, utilizzato all'interno di questa tesi.

Oltre a consentire la raccolta di dati che permettano il riconoscimento di azioni umane, focus principale di questo lavoro, tali set possono essere utilizzati in altri campi. Uno dei campi è monitorare il movimento e gli spostamenti in tempo reale e fornire feedback immediato all'utente consentendo di correggere postura e tecnica di movimento durante l'attività fisica o durante il lavoro, migliorando le prestazioni e prevenendo gli infortuni.

1.4 Vantaggi e svantaggi di IMU e RGB-D

Le videocamere RGB-D sono molto utilizzate nel settore del HAR visti i costi relativamente bassi, anche se molto maggiori rispetto a quelli delle semplici videocamere RGB, e la capacità di riconoscere il posizionamento di eventuali og-

getti all'interno della scena. Fornendo così utili informazioni da integrare alla posizione del corpo.

L'utilizzo di videocamere presenta, tuttavia, lo svantaggio di costringere l'utilizzo ad una zona limitata, in cui è necessario un posizionamento tale da evitare occlusioni della persona nell'immagine da parte di altri oggetti. Motivo per cui, in luoghi chiusi con numerosi oggetti che possono ostacolare la visione, gli IMU sono più consigliati. Inoltre, ToF e luce strutturata non funzionano sotto esposizione diretta alla luce solare o con illuminazione non uniforme, escludendone così l'utilizzo all'aperto o in situazioni di illuminazione particolare.

Le videocamere RGB-D richiedono una calibrazione accurata per ottenere risultati precisi. Tuttavia, la calibrazione può essere difficile da eseguire correttamente, soprattutto se la fotocamera è soggetta a movimenti o se viene utilizzata in condizioni di luce mutevoli. Inoltre, nel caso tali fenomeni siano particolarmente forti posso portare a richiedere anche frequenti calibrazioni nel tempo.

Gli IMU sono generalmente dispositivi indossabili, che l'operatore pone sul proprio corpo in posizione predefinite. Tali sensori non necessitano quindi di una posizione fissa nell'ambiente come per telecamere, e sono di conseguenza utilizzabili in situazioni di qualsiasi tipo. Non vengono, infatti, influenzati da fenomeni come luce variabile o la presenza di oggetti. Per questi motivi possono essere utilizzati anche all'esterno e durante la vita quotidiana. Non utilizzando immagini consentono, inoltre, di monitorare maggiormente la privacy dell'utente.

Ulteriormente, gli IMU possono essere utilizzati per registrare il movimento di parti del corpo difficili da monitorare con altri metodi, come ad esempio alcune vertebre. Ciò consente di ottenere maggiori informazioni sulla cinematica del movimento e sull'interazione tra diverse parti del corpo. In termini di precisione, ci si aspettano dei risultati più accurati da parte degli IMU rispetto alle immagini. Principalmente perché le immagini non sono in grado di catturare movimenti di lieve entità. Ad esempio, una rotazione dell'avambraccio mentre si sta stringendo una vite può risultare difficile da cogliere da parte delle videocamere, ma risultare facilmente percettibile da parte di un sensore IMU posto sul braccio. Si presuppone in generale una capacità di riconoscimento delle azioni maggiore in base a quanto più sono dettagliati i movimenti descritti dagli scheletri raccolti; data la maggior precisione degli scheletri di tipo inerziale, si attende una maggiore precisione in fase di riconoscimento da parte dei sensori inerziali.

Tuttavia, gli IMU possono essere soggetti ad errori di misura e drift, soprattutto nel lungo termine, il che può richiedere una calibrazione regolare per

garantire la precisione dei dati rilevati. Inoltre, IMU di alta precisione possono essere costosi, soprattutto se si vuole un errore molto basso. Per ottenere una maggiore precisione cominciano anche ad aumentare altre caratteristiche come dimensioni e peso, che provocano grossi problemi vista la necessità di essere indossati da operatori umani per un lungo lasso temporale. In alcuni casi, altri dispositivi elettronici possono provocare interferenze risultanti in errori di misurazione. Ulteriore svantaggio degli IMU è la necessità di dover avere una fonte di alimentazione che segua la persona, che comporta quindi l'utilizzo di batterie o eventualmente di cavi, che risultano però particolarmente scomodi. Mentre, le telecamere posso essere facilmente collegate alla rete elettrica, non necessitando così di batterie.

1.5 HAR basato su scheletri

Oltre ai metodi discussi, esiste una modalità di registrazione dei dati che dipende dalle tecnologie citate. Si tratta dell'estrazione degli scheletri, ovvero le posizioni di giunti del corpo (gomito, polso, spalla, ginocchia, etc), a partire da dati di videocamere RGB-D o di IMU.

Il vantaggio di usare uno scheletrizzatore è che fornisce una rappresentazione accurata, ma al tempo stesso leggera, dei movimenti di una persona. Caratteristica che consente di ridurre i costi computazionali e i tempi necessari quando si effettua il riconoscimento di azioni.

Figura 1.4: Stima della posizione del corpo un'immagine RGB [4].

Inoltre, in tutti i casi viene preservata la privacy dell'utente dato che non diventa necessario diffondere immagini. Dalla proprietà di non utilizzare immagini ne consegue che si tratti di una soluzione robusta contro le variazioni di background e di pattern nei vestiti.

La rappresentazione tramite lo scheletro fornisce un livello di astrazione maggiore dei movimenti umani, se comparato ai dati grezzi degli IMU, semplificando di conseguenza l'analisi e riducendo la complessità dei dati. Inoltre, si aumenta la

resistenza a diversi posizionamenti dei sensori. Il posizionamento potrebbe variare tra individui e scenari. Ciò diventa un problema quando si considerano i dati di accelerometri e altri sensori, ma non per lo scheletro, che dopo la calibrazione dovrebbe risultare particolarmente resistente a questo fenomeno.

Infine, lo scheletro non risulta essere disturbato da processi come traslazione e rotazione. Di conseguenza, nonostante la posizione e l'orientamento della persona, la rappresentazione scheletrica continua a catturare in maniera accurata il movimento relativo tra i giunti. Mentre i dati grezzi degli IMU ne risentirebbero di queste trasformazioni globali.

Capitolo 2

Stato dell'arte

Vista l'importanza a livello tecnologico e industriale dell'HAR sono numerosi i lavori che hanno trattato l'argomento. Il punto chiave di questo capitolo è una breve analisi dei risultati raggiunti da questi. Portando poi una breve panoramica di alcuni dei dataset più popolari. Infine, viene fatta una breve parentesi sulla classificazione di un segnale tramite un MLP.

2.1 Confronto tra videocamere RGB-D e IMU

2.1.1 Precisione delle misurazioni

A conferma del principio su cui si basa l'idea per cui i sensori inerziali dovrebbero risultare più precisi nell'acquisizione rispetto a videocamere RGB-D viene portata l'analisi di Nicolas Valencia-Jimenez in [11]. Tale studio dimostra che nel caso della misurazione dell'angolo di un giunto del corpo un sistema composto da 2 videocamere RGB-D Microsoft Kinect si ha un errore nella misurazione con media di 4.9° e valore massimo di 9°. Valori che hanno migliorato, grazie all'utilizzo di un sistema composto da più videocamere RGB-D, il risultato di 14,6° di errore trovato da Tannous et al. [12] con un singolo dispositivo. Si ha invece, che gli IMU MTw di Xsens hanno ottenuto un errore medio di solamente 3.7°, dimostrando così la maggior precisione da parte di questa tipologia di dispositivi.

2.1.2 Human Action Recognition

Lo studio di Luis Roda-Sanchez et al. [13] va ad analizzare l'utilizzo di immagini RBG-D e IMU nel contesto di riconoscimento delle gesture per il controllo di robot da utilizzare nel remanufacturing, ovvero il processo di ricostruzione

di un prodotto secondo le specifiche del prodotto fabbricato originale utilizzando una combinazione di parti riutilizzate, riparate e nuove. Trattandosi di un processo che richiede forte flessibilità diventa necessaria un efficace interazione uomo-robot. Dall'utilizzo di dispositivi RGB-D e IMU i risultati evidenziano che tramite sensori inerziali si ottiene un risultato fino a 8.5 volte migliore rispetto alla controparte. I riconoscimenti da parte dei sensori RGB-D si sono, infatti, dimostrati dipendenti dal piano di esecuzione del movimento e la postura del soggetto. Sono poi emerse ulteriori conclusioni. L'utilizzo di IMU per la caratterizzazione del movimento richiedono risorse hardware meno potenti per ottenere informazioni 3D da dati bidimensionali. Inoltre, i dispositivi wearable richiedono di essere alimentati da batterie esterne, necessitando di algoritmi di raccolta dei dati particolarmente ottimizzati e di protocolli di comunicazione a bassa energia. Mentre i sistemi di visone posso facilmente essere collegati alla rete elettrica.

Invece, Ong Chin Ann e Lau Bee Theng in [14] tramite un'analisi di 32 articoli scientifici determinano che nella scelta tra sensori di profondità e sensori indossabili non esiste un vincitore assoluto. Non esiste una misura che possa misurare tutti i parametri che compongono il contesto globale. Infatti, ciascuna tipologia di sensori ha i suoi punti di forza e di debolezza. Bisogna quindi fare un'analisi del contesto di utilizzo per determinare quale tipologia sia più indicata. Infine, lo stesso lavoro determina che gran parte dei ricercatori utilizzano i sensori Microsoft Kinect per la parte sperimentale di HAR grazie ai bassi costi e all'elevata efficienza.

2.2 Riconoscimento di azioni basato su scheletri

Grazie alla leggerezza dei dati e la grande quantità di informazioni portate da parte dei giunti del corpo umano quando si effettua un movimento, l'utilizzo delle sequenze di scheletri è stato largamente approfondito da parte della letteratura.

Varie reti di deep learning ed approcci sono stati proposti in letteratura per sfruttare al meglio il tipo di dato costituito dagli scheletri 3D, ad esempio:

- RNN - Recurrent neural network
- CNN - Convolutional neural network
- GNN - Graph neural network
- Transformer based

Le RNN e le loro varianti, come ad esempio le Long short-term memory (LSTM), sono in grado di imparare le dipendenze tra dati sequenziali o dati di serie temporali. Si distinguono per la loro memoria, in quanto prelevano informazioni dagli input precedenti per influenzare l'output relativo all'input corrente. Mentre le reti neurali profonde tradizionali presuppongono che input e output siano indipendenti l'uno dall'altro, l'output delle reti neurali ricorrenti dipende dagli elementi precedenti all'interno della sequenza [15]. Tra i vari lavori [16] [17] [18] esistono anche varianti che cercano di ottenere risultati migliori tramite tecniche particolari. Ad esempio [19] propone di dividere lo scheletro umano in 5 parti, fornire in modo separato ciascuna parte ad una RNN e unire poi gli output in modo gerarchico.

Le CNN [20] sono un tipo di modello di deep learning per il processamento di dati che hanno un pattern a griglia, come le immagini. Sono progettate con lo scopo di imparare automaticamente da dati con feature nel dominio dello spazio, partendo da pattern di basso livello e arrivando all'alto livello. Tuttavia, quando vengono utilizzate per il riconoscimento di azioni tramite scheletro il modellamento dell'informazione spazio temporale diventa un problema. Nascono quindi delle soluzioni alternative per la trasformazione di sequenze di scheletri in immagini. Ad esempio, nell'articolo [21] viene posto sull'asse x l'asse temporale e sull'asse y le posizioni dei vari giunti, come riportato in Figura 2.1. Creando così una singola immagine che racchiuda tutte le informazioni.

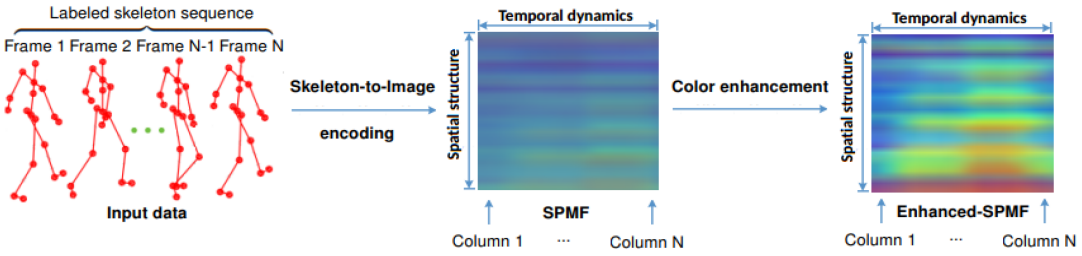


Figura 2.1: Trasformazione di una sequenza di scheletri in una singola immagine [21]

Le GNN consentono di sfruttare al meglio la natura degli scheletri. Questo perché per natura i dati dello scheletro risultano formare un grafo. Ne consegue che numerosi lavori hanno sperimentato questa tipologia di modello ed eventuali varianti [22] [23] [24] [25].

Infine, le Transformer [26] sono state sviluppate per risolvere il problema delle Sequence transduction, ovvero la trasformazione di una sequenza di input in una

sequenza di output, meccanismo che richiede la capacità di memoria degli ingressi precedenti. Queste vanno a migliorare le RNN introducendo il meccanismo di Attenzione [27], la capacità di dare maggiore importanza a determinati fattori. Alcuni lavori che utilizzano le Transformer sono [28], [29] e [30].

2.2.1 Risultati

Nella Tabella 2.1 vengono riportati i lavori, tra quelli trovati, che hanno portato a risultati migliori sugli scheletri contenuti nei dataset NTU RGB+D [31] e NTU RGB+D 120 [32]. In tale tabella con cross subject si indica la validazione con soggetti diversi, possibile grazie all'elevato numero di partecipanti presenti nel dataset. Mentre con cross view viene indicato l'utilizzo di punti di vista diversi dello scheletro. Infine, con cross setup s'intende l'utilizzo di campioni con ID del setup pari per il training e l'utilizzo di campioni con ID del setup dispari per il testing. A ciascun ID del setup corrispondono un'altezza dei sensori e una distanza dal soggetto differenti.

METODO	**ARTICOLO**	**ANNO**	**DATASET**			
			NTU RGB-D		**NTU RGB-D 120**	
			cross subject	**cross view**	**cross subject**	**cross setup**
RNN	[18]	2021	97.0	98.5	90.6	86.7
CNN	[33]	2020	84.2	89.7	74.8	76.9
GNN	[25]	2022	93.0	97.1	89.8	91.2
Transformer	[29]	2022	92.3	96.5	88.3	89.2

Tabella 2.1: Risultati migliori trovati con metodi skeleton based.

2.3 Dataset

Alcuni dei dataset principali risultano essere: NTU RGB+D [31], NTU RGB+D 120 [32], UWA 3D Multiview Activity II [34], UTD-MHAD [35], DIP-IMU [36], CIP [37].

2.3.1 Dataset RBG-D e scheletri

Nel caso di immagini RGB, RGB-D e dati 3D riguardanti gli scheletri spiccano NTU RGB+D e NTU RGB+D 120. Come dichiarato dal ROSE Lab [38], NTU RGB+D contiene 60 classi di azioni e 56880 campioni video. Invece, NTU

RGB+D 120 estende NTU RGB+D aggiungendo altre 60 classi e altri 57600 campioni video. Di conseguenza, NTU RGB+D 120 ha 120 classi e 114480 campioni in totale. Entrambi i dataset contengono per ogni campione: video RGB, mappe di profondità, dati 3D degli scheletri e video infrarossi (IR). Tutti catturati da 3 dispositivi Kinect V2. La risoluzione dei video RGB è 1920x1080, mentre le mappe di profondità e i video IR sono 512x424. I dati degli scheletri 3D contengono 25 giunti del corpo per ogni frame.

Ma si tratta di solo alcuni dataset con queste caratteristiche, sono infatti numerosissimi. Ad esempio è presente il dataset UTD Multimodal Human Action Dataset (UTD-MHAD) [39] [35]. Quest'ultimo contiene 27 azioni eseguite da 8 soggetti (4 maschi e 4 femmine). In cui ciascuna azione viene eseguita 4 volte registrando di tutti i campioni video RGB-D, video con profondità, posizioni dei giunti e il segnale da parte di un singolo sensore inerziale. Vengono poi segnalati: UWA 3D Multiview Activity II [34], ETRI-Activity3D [40], N-UCLA [41], ecc.

2.3.2 Dataset IMU

Per i sistemi composti da più IMU la ricerca di dataset si dimostra molto più difficile e con una produzione di risultati più scarsi. Infatti, come evidenziato anche da Manuel Palermo et al. in [37], la maggior parte dei dataset utilizza smartphone [42] o singoli dispositivi collegati a singoli arti [35]. Di conseguenza vengono presentate dinamiche semplici nel caso del coinvolgimento di tutto il corpo, in cui i partecipanti si concentrano principalmente sul movimento della parte interessata [43] [44].

Ad evidenziare l'assenza di dataset su dati inerziali che coinvolgano l'intero corpo, Huang et al. in [36] arrivano a ricavare dei dati che simulino degli IMU, utilizzando dataset di scheletri ottenuti da immagini RGB-D. Va comunque fatto notare che tali dati non contengono caratteristiche tipiche di sensori reali come, ad esempio, rumore e dati dei magnetometri. Motivo per cui successivamente è stato realizzato il dataset Deep Inertial Poser IMU (DIP-IMU) da parte degli stessi. Quest'ultimo è stato creato tramite 10 soggetti con addosso 17 sensori MTw Awinda di Xsens. Sono state registrate 64 sequenze, producendo materiale per un totale di 92 minuti e 330178 frame.

Anche il dataset Complete Inertial Pose (CIP) Dataset [37] nasce dalla problematica di avere dataset realativi ad IMU con pochi dati e concentrati su un'unica sezione del corpo.

2.4 Dataset utilizzato

Prendendo spunto dai dataset della letteratura e dalle relative procedure di acquisizione, in questo lavoro di tesi è stato acquisito un dataset di IMU e dati RGB per 3 soggetti e 5 ripetizioni, focalizzandosi su 10 azioni di tipo relativo ad attività motoria e HRC. I soggetti che hanno preso parte alla raccolta dati sono tutti maschi di età e statura differenti.

2.5 Classificazione di un segnale tramite MLP

In [45] Yuan-Pin Lin et al. vanno a riconoscere il tipo di canzone in esecuzione durante la registrazione di diversi segnali provenienti da Elettroencefalogramma. Le tipologie di canzoni hanno lo scopo di provocare delle emozioni in modo da produrre determinati segnali da parte del cervello. Questo lavoro risulta interessante in quanto utilizza un Multilayer Perceptron per la classificazione di segnali, passando attraverso la trasformata di Fourier. Si tratta di un approccio simile a quanto verrà presentato nel Capitolo 4.

Mentre [46] utilizza un MLP per la classificazione di azioni a partire dai raw data provenienti da singole IMU posizionate su braccia o gambe.

Capitolo 3

Metodologia

In questo capitolo si vogliono analizzare le basi per quanto riguarda il riconoscimento delle azioni umane. Per fare ciò si definisce anche il più generale concetto di problema di classificazione. Negli anni sono state sviluppate numerose soluzioni a questo problema, le quali possono essere utilizzate singolarmente o unite. Motivo per cui il focus di questo lavoro si pone principalmente sulle soluzioni che vengono effettivamente utilizzate in parte o totalmente durante la fase sperimentale.

In genere, il riconoscimento di azioni consiste nel predire quale azione è in corso rispetto un insieme predefinito di possibili azioni. Si tratta quindi di un problema di classificazione, in cui dato un input si cerca di predire quale delle possibili classi di interesse (azioni in questo caso) è quella effettivamente più probabili per il dato input. Numerosi articoli, tra cui [47] e [48], propongono una divisione delle fasi del processo di riconoscimento di azioni. In generale la struttura può essere separata in 4 fasi, che verranno maggiormente descritte nel seguito di questo capitolo:

- Acquisizione dei dati: raccolta di dati che coinvolgono azioni umane, provenienti da uno o più tipi di sensori.

- Pre-processing: i dati grezzi che sono stati acquisiti vengono sottoposti a delle operazioni di filtraggio e campionamento allo scopo di ridurre il rumore e le dimensioni dei dati. L'obiettivo finale è quello di eliminare dati indesiderati che possono influire su prestazioni e precisione.

- Estrazione feature: è l'estrazione di caratteristiche importanti al fine della classificazione.

- Classificazione: a partire da un input si cerca di definire l'appartenenza a una classe tra quelle disponibili, producendo quindi come output la classe che viene determinata come più affine.

3.1 Riconoscimento di azioni umane

Si tende spesso a fare confusione tra i termini gesto, azione, interazione e attività di gruppo. Nella maggior parte dei lavori scientifici [49] vengono distinte 4 principali categorie:

- Gesto: movimento elementare di una parte del corpo di una persona, ad esempio alzare un braccio.
- Azione: movimento compiuto da una singola persona e composto da più gesti, implicando quindi il movimento di più parti del corpo. Alcuni esempi sono camminare e saltare.
- Interazione: si tratta di un azione in cui una persona interagisce con altre persone o oggetti. Appartengono alle interazioni situazioni come un abbraccio e il lancio di un oggetto.
- Attività di gruppo: si tratta di azioni compiute da gruppi concettuali di più persone o più oggetti. Risulta essere la categoria più complessa e vi appartengono esempi come un gruppo di persone che marcia o che seguono un incontro.

Il focus di questa tesi si pone sul riconoscimento di azioni, in quanto vengono considerati un solo soggetto e movimenti composti da più gesti. Motivo per cui con HAR si andrà sempre ad intendere il riconoscimenti di azioni umane (Human Action Recogniton). Va comunque detto che spesso nella letteratura si tendono ad equiparare Human Action Recognition e Human Activity Recognition, facendogli assumere lo stesso significato.

3.2 Acquisizione dei dati

Nell'acquisizione dei dati si possono raccogliere informazioni con mezzi di natura diversa. Le fonti principali per quanto concerne i dati utilizzati nel HAR sono: RGB, scheletro, profondità, infrarossi, point cloud, event stream, audio, accelerazione, radar o WiFi. Che possono essere unite in modo da ottenere informazioni

più precise. Un confronto tra le modalità citate e tra le loro combinazioni vengono presentate da Zehua Sun et al. [1]. Va fatto notare che spesso la combinazione può avvenire anche nella fase successiva all'acquisizione. Ad esempio, creando due modelli che producano un risultato finale di cui si effettua successivamente la media. Si parla in tal caso di Score Fusion.

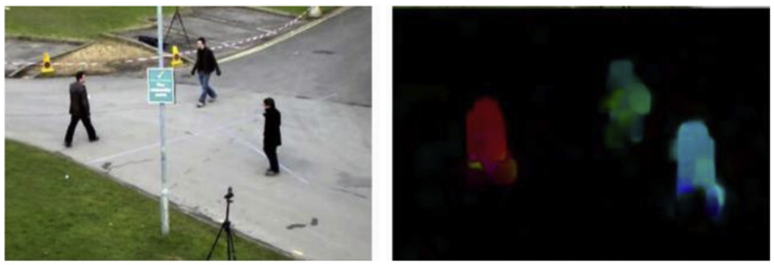
3.3 Pre-processing

Il pre-processing consiste nella preparazione dei dati raccolti in un formato che porti alle migliori prestazioni possibili in termini di tempo necessario per la classificazione e in precisione nella classificazione. A questo scopo vengono effettuate operazioni di riduzione dei rumori e della dimensione dei dati. Le modifiche comportate dal pre-processing possono portare a un cambiamento leggero dei dati o anche a un cambiamento totale della loro natura. Alcune tecniche possibili di preprocessing sono: la trasformazione di video RGB in immagini, la gestione dei dati mancati e la normalizzazione.

3.3.1 Trasformazione video RGB in immagini

Un esempio di pre-processing può essere considerato, per quanto concerne il riconoscimento di azioni tramite modalità RGB, la trasformazione di video in una singola immagine. La modifica lavora su sequenze di immagini catturate da un'unica videocamera RGB. La tecnica risulta interessante principalmente per il largo bacino di applicazioni, tra cui la video sorveglianza. Ulteriore punto a favore per questa tipologia di riconoscimento è la semplicità di utilizzo, vista la necessità di una singola videocamera ormai disponibile a costi bassissimi. Tuttavia, porta con se numerose difficoltà per chi cerca di sviluppare delle predizioni corrette. Tra le difficoltà maggiori fenomeni come occlusione della vista a causa di oggetti, variazioni nelle condizioni di illuminazione e costi computazionali particolarmente alti a causa delle dimensioni delle immagini.

La sfida più grande per l'utilizzo di sequenze d'immagini è stato riuscire a rappresentare l'asse temporale. Motivo per cui ci sono nella letteratura numerosi lavori riguardo la condensazione di video in immagini statiche [6] (Figura 3.1). Questi lavori presentano, però, il difetto di portare ad una perdita di alcune informazioni temporali che possono risultare fondamentali per il corretto riconoscimento di specifiche azioni. Motivo per cui non verranno trattati più approfonditamente.



(a) (b)

Figura 3.1: Condensazione di un video RGB in un immagine [6]

3.3.2 Gestione dati mancanti

In alcuni casi si può verificare il fenomeno dei dati mancanti. Problematica che viene risolta in fase di pre-processing. Ci sono diverse tecniche utilizzate per gestire questi dati, come la rimozione dei campioni in cui sono assenti i dati, la sostituzione dei valori mancanti con la media o la mediana, oppure l'uso di tecniche più sofisticate. Sul tema si rimanda ad esempio all'articolo di Tahera Hossain e Sozo Inoue [50], in cui viene trattata l'assenza di dati specificatamente nel HAR.

3.3.3 Normalizzazione

La normalizzazione è importante per molte reti neurali. Si tratta di un ridimensionamento utilizzato solitamente quando una caratteristica assume valori particolarmente differenti, ad esempio tra 0.001 e 1000. Lo scopo è di avere un range più ristretto e una scala che diano risalto anche ai valori bassi.

I due metodi più comuni di normalizzazione [51] sono:

- Min-max:

$$v' = \frac{v - min_A}{max_A - min_A} (new\_max_A - new\_min_A) + new\_min_A$$

- Z-score:

$$v' = \frac{v - mean_A}{stand\_dev_A}$$

(dove v è il valore precedente e v' il nuovo valore).

3.4 Estrazione feature

Le feature sono delle caratteristiche che vengono usate come input per un classificatore. Con estrazione delle feature s'intende il processo di trasformare le caratteristiche, più importanti allo scopo della classificazione, in dati di natura numerica.

Tali feature possono essere estratte in due modi: manualmente, determinando particolari parametri che risultano essere utili per il riconoscimento, o automaticamente, tramite delle reti neurali. Si vedrà nella fase di classificazione che questa differenza risulta essere uno dei punti chiave per la differenziazione tra algoritmi di Machine learning e algoritmi di Deep learning.

3.5 Classificazione

Spesso si fa confusione tra i termini Intelligenza artificiale (AI), Machine learning (ML), Deep learning (DL) e Neural Network (NN). Vengono, infatti, usati comunemente in modo intercambiabile nelle conversazioni, nonostante presentino differenze sostanziali.

Prima di analizzare più nello specifico le differenze bisogna fare una precisazione. Con Neural network a seconda del contesto ci si può riferire a: Biological neural network (BNN) o Artificial neural network (ANN). La prima risulta essere la rete neurale che si trova ad esempio nel cervello umano. La seconda si riferisce, invece, a tecnologie che in parte vanno a simulare alcune caratteristiche del cervello umano all'interno di computer. Durante questa tesi verrà dato per scontato che con NN ci si riferisca alle ANN.

Il Machine learning è una branca dell'Intelligenza artificiale. Alcuni algoritmi di Machine Learning vanno ad utilizzare le Neural network. Nel caso il numero di layer che compongono la Neural network sia elevato si parla di Deep learning, quindi si tratta di una specializzazione del Machine learning. Esistono comunque algoritmi di Machine learning che non appartengono al Deep learning, ma che usano comunque delle reti neurali, seppure con un limitato numero di layer.

Le caratteristiche che rendono gli algoritmi di Deep learning così particolari rispetto agli altri tipi di algoritmi, che appartengono comunque alla categoria di Machine learning, sono:

- struttura dell'algoritmo della NN;
- minor richiesta di intervento umano;

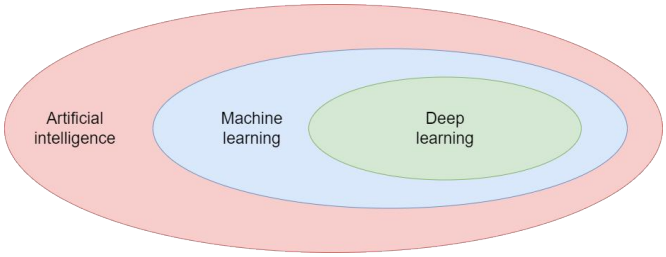


Figura 3.2: AI, ML e DL.

- quantità di dati richiesti.

La minor richiesta di intervento umano si ha prima di tutto nella fase di estrazione delle feature. Negli algoritmi di DL questa fase viene effettuata dalla rete neurale, mentre in Machine learning al di fuori del Deep learning viene richiesta l'azione umana. Questa differenza viene evidenziata in Figura 3.3. Successivamente, gli algoritmi di Deep learning possono effettuare una classificazione anche di dati non etichettati. Caratteristica che da vita alla distinzione tra algoritmi supervisionati, ovvero che usano dati etichettati per il training, e non supervisionati, cioè che lavorano con dati non etichettati anche nella fase di training.

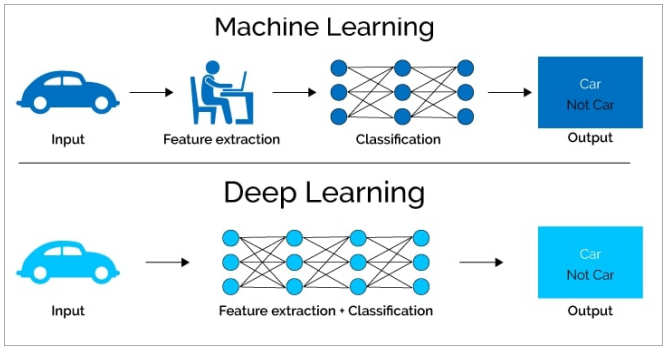


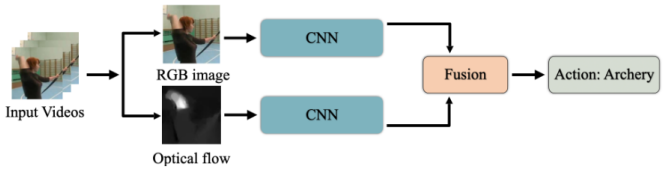
Figura 3.3: Una delle differenze tra DL e ML. [52]

Successivamente, ciò che rende un algoritmo di Machine learning un algoritmo di Deep learning è l'utilizzo di una rete neurale che presenti più di 3 layer, inclusi input layer e output layer.

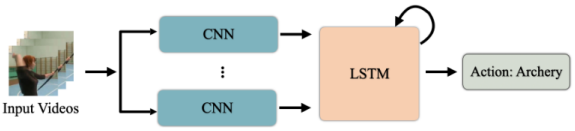
Poiché il ML è una tecnica che impara delle relazioni input/output, in base ad una serie di esempi forniti, per il corretto funzionamento gli algoritmi necessitano di grosse quantità di dati. La richiesta di dati aumenta soprattutto nel caso non siano etichettati.

Negli ultimi anni il Deep learning sta crescendo a dismisura, nonostante le problematiche di necessitare di elevata potenza di calcolo e di una grossa mole di dati. Tuttavia, grazie allo sviluppo di GPU ad alte performance in congiunzione

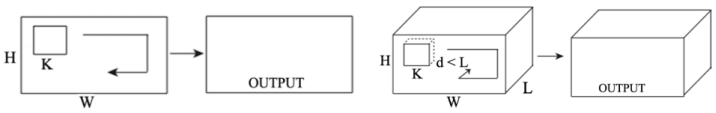
con il miglioramento del cloud computing, il DL sta spopolando. Ulteriore mezzo di allegerimento rispetto alle richieste del ML è la nascita di tecniche come il trasfer learning, una tecnica in cui si sfruttano reti allenate per task simili a quello richiesto in modo da non dover far partire l'allenamento della rete da zero, con relativo risparmio di risorse.



(a) Schema of two-stream 2D CNN-based methods.



(b) Schema of LSTM-based methods.



(c) Schema of 2D convolution. (d) Schema of 3D convolution.

Figura 3.4: Alcuni esempi di metodi utilizzati nella classificazione [53].

Gli algoritmi di ML e di DL più utilizzati nell'ambito del riconoscimento di azioni umane sono [53]: Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short Term Memory (LSTM). Svariati esempi vengono riportati nel Capitolo 2. In seguito si approfondisce il MLP, in quanto utilizzato nella fase sperimentale.

3.5.1 Multilayer Perceptron - MLP

Per spiegare il funzionamento di un Multilayer Perceptron si utilizza un semplice esempio del riconoscimento di un numero disegnato a mano. La Figura 3.5 è un numero scritto su un riquadro 28x28 pixel in cui è stato disegnato un 1 e si vuole che il computer sia in grado di riconoscerlo. Si ricorre quindi al Multilayer Perceptron.

La prima cosa da definire è il concetto di neurone. Ciascun neurone può essere visto come un numero compreso tra 0 e 1. Tale valore è detto attivazione del neurone. Tutta l'informazione passa attraverso i neuroni, di conseguenza bisogna tradurre l'immagine di partenza in una struttura che possa essere contenuta nei

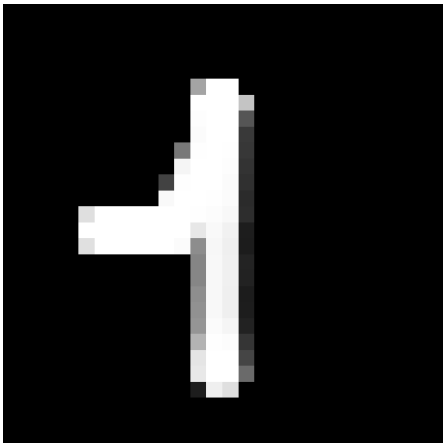


Figura 3.5: Numero scritto a mano.

neuroni. Si crea un neurone per ogni pixel e si traduce il colore di quel pixel in un valore compreso tra 0 e 1. Si ottengono così $28 * 28 = 784$ neuroni. E' stato in questo modo creato il layer di input. Sapendo che le cifre scritti possono essere da 0 a 9 si può già creare l'output layer, composto da 10 neuroni. Il neurone che alla fine di ciascuna esecuzione avrà il valore più alto sarà la predizione della rete neurale. Viene così risolto un problema di classificazione, dove le cifre da 0 a 9 sono le possibili classi da riconoscere

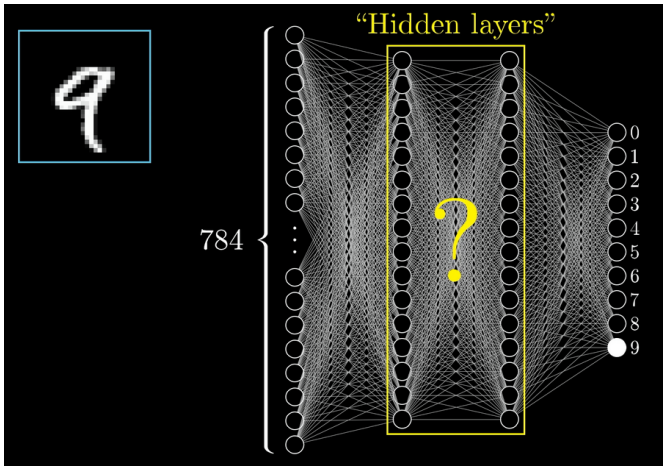


Figura 3.6: Struttura MLP con hidden layers [54].

Ciò che consente di passare dall'input layer all'output layer sono gli hidden layer. Si tratta di una struttura che viene determinata anche tramite la sperimentazione. Ogni neurone degli hidden layer è connesso a tutti i neuroni del layer successivo e tutti i nodi del layer precedente. Ogni collegamento ha un peso, che indica quanto il neurone precedente influisce sull'attivazione del neurone successivo. Infatti, il valore contenuto all'interno di ciascun neurone viene ottenuto

facendo la somma dei prodotti tra attivazione del nodo precedente e il peso della connessione. Siano $w_1, w_2, w_3, ..., w_N$ i pesi dei collegamenti e $a_1, a_2, a_3, ..., a_N$ i valori di attivazione dei neuroni collegati ad un singolo neurone del layer successivo con attivazione a'. Allora

$$a' = \sum_{i=1}^{N} a_i * w_i$$

Il valore ottenuto probabilmente non sarà compreso tra 0 e 1. Bisogna quindi utilizzare una funzione che riporti il valore nell'intervallo desiderato. Si utilizzando quindi le Activation function. Un esempio è la Funzione sigmoide:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Nelle applicazioni moderne viene utilizzata la ReLu (rectified linear unit), nonostante non presenti bound superiore, definita come segue:

$$ReLu(x) = \begin{cases} x & x > 0 \\ 0 & altrimenti \end{cases}$$

In alcuni casi si può non volere che il neurone si accenda appena la somma pesata supera 0. Motivo per cui si aggiunge un numero detto bias. Nella pratica si aggiunge alla somma pesata questo numero. Si ottiene quindi che:

$$a' = \sigma(\sum_{i=1}^{N} a_i * w_i + b')$$

Ad esempio, si può richiedere che si accenda quando la somma pesata supera 10. Per ottenere questo risultato basta porre $b' = -10$. Il processo descritto viene fatto per tutti i neuroni dei vari layer. Si hanno quindi moltissimi parametri da poter modificare, che vengono inizializzati in modo casuale. Successivamente, si fornisce al modello un'immagine proveniente dai dati di training. Viene effettuato un confronto nell'output layer tra il risultato ottenuto e il risultato corretto, fornito sotto forma di annotazione assieme ai dati. Si tratta, infatti, di allenamento supervisionato in cui alla rete sono mostrati esempi di input e output da cui imparare la relazione sottostante. Al fine del confronto tra il risultato corretto e il risultato ottenuto viene definita una Cost function, ad esempio la somma dei quadrati delle differenze tra layer ottenuto e layer atteso. Ciò che si vuole è però la media di questa Cost function su tutti i training samples forniti. L'obiettivo diventa trovare i pesi e i bias che consentano di minimizzare la media delle Cost

function. Ciò si ottiene tramite il Gradient descent.

Visto che la media della Cost function su tutti i training samples impiegherebbe molto tempo per essere calcolata si usa spesso il Stochastic gradient descent, in cui si fa la media solo su porzioni del dataset, dette Batches. Su tale porzione vengono effettuati più passaggi da parte del modello (epoche). Ad ogni iterazione il modello MLP modifica i valori dei pesi e dei bias, arrivando infine a trovare l'insieme di valori "ottimo" che minimizza la loss function. Tale insieme di valori costituiscono il modello allenato che può poi essere utilizzato per fare predizioni su nuovi dati.

Capitolo 4

Strumenti

L'obiettivo di questo capitolo è introdurre gli strumenti fondamentali utilizzati nella parte sperimentale del lavoro. Ciò allo scopo di garantire una miglior comprensione del funzionamento di tutte le piattaforme software, e non, che hanno consentito di portare a termine gli esperimenti.

In particolare, per la realizzazione del setup sperimentale sono stati utilizzati i sensori inerziali Xsens con il relativo codice proprietario, ed una rete di telecamere basata sul progetto open-source OpenTrack per la stima della posa delle persone. Per entrambe le tipologie di dati, inerziali e visivi, è stato utilizzato il sistema ROS come interfaccia comune con cui gestire i sensori e l'acquisizione dei dati. Infine, con i dati raccolti è stato allenato una rete di machine learning per il riconoscimento di azioni utilizzando il tool online Edge Impulse. Nel resto del capitolo ciascuno di questi moduli software è descritto nel dettaglio.

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

Figura 4.1: I quattro strumenti utilizzati.

4.1 ROS - Robot Operating System

ROS (Robot Operating System) è un software middleware, ovvero un insieme di librerie software e tool, ampiamente utilizzato in robotica. Grazie al suo design

fortemente modulare che si presta al riuso del codice e alla facilità di utilizzo, dato che per scrivere codice si utilizzano linguaggi come Python e C++, è diventato particolarmente popolare. Tanto da essere utilizzato da aziende come la NASA [55].

Esistono due versioni principali di ROS: ROS1 e ROS2. E ciascuna versione presenta più distribuzioni. In questo lavoro è stata utilizzata la distro Noetic Ninjemys di ROS1, per Ubuntu. Il motivo fondamentale alla base di tale scelta è la stabilità del software.

Il concetto chiave dietro ROS è avere una rete di nodi connessi tra di loro, che s'inviano messaggi attraverso dei canali appositi detti topic. I vari topic e nodi vengono gestiti e registrati da un nodo unico e speciale, il master. Ciascun topic presenta un proprio nome e il tipo di dati che vengono inviati attraverso sono definiti in modo specifico tramite i messaggi.

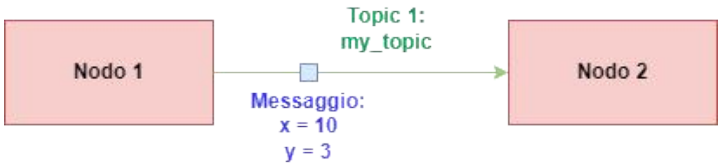


Figura 4.2: Struttura ROS per i nodi, topic e messaggi

4.1.1 Messaggi

I messaggi sono un formato serializzato per dati strutturati. Che consentono di far comunicare nodi C++ con nodi Python. Il formato viene descritto nei file di formato .msg, contenenti le definizioni di nomi e tipi per ciascun campo. Tali tipi possono essere primitivi, come ad esempio char, int e float, ma anche essere array o strutture complesse.

4.1.2 Master

Il master, come già detto, è un nodo particolare che ha il compito di registrare l'indirizzo di rete degli altri nodi e altre informazioni. Consente di informare dei nodi subscriber su quali nodi publisher stanno pubblicando sopra un determinato topic. Motivo per cui tutti i nodi devono conoscere l'indirizzo di rete del nodo master, che deve rimanere invariato (di default il port del master è 11311).

In questo modo quando un nodo vuole iscriversi ad un determinato topic gli basta specificare il nome e inviare una richiesta al master con quest'ultimo. Il

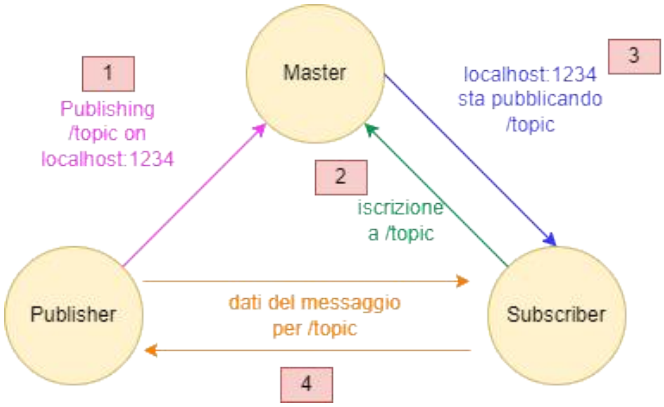


Figura 4.3: Funzionamento di una struttura con un subscriber, un publisher e un master

master restituirà il port su cui viene pubblicato un determinato topic, in modo che il nodo che ha effettuato la richiesta possa leggere i dati che vi sono scritti sopra.

4.1.3 Bag

ROS mette a disposizione dei tool e dei file che consentono di raccogliere i messaggi inviati in un certo periodo di tempo e di effettuare la riproduzione successivamente. Ripubblicando i messaggi sui topic in corso durante la registrazione. Il tool prende il nome di rosbag, mentre i file utilizzati sono in formato .bag .

4.1.4 RViz

RViz è il tool più famoso messo a disposizione da ROS. Permette di visualizzare i dati, che seguono alcune tipologie di messaggi ammessi dal tool (come immagini, dati 3D, etc), che vengono pubblicati sui topic.

4.2 OpenPTrack

OpenPTrack [56] è un progetto open source con lo scopo di trovare una soluzione al tracciamento umano tramite una rete di telecamere. L'idea di base è quella di tracciare la posizione di più persone su grandi aree e in tempo reale.

Si basa sul middleware ROS e offre: una calibrazione user friendly, rilevamento di più persone da immagini RGB/ad infrarossi/di profondità, output in streaming UDP a 30fps.

Per portare a termine il compito di rilevamento è necessario che ogni videocamera sia connessa ad un computer su cui giri la distro Ubuntu di Linux. Inoltre, ciascun computer deve essere connesso ad una rete Ethernet Gigabit. I dati raccolti sono quindi inviati ad un processo master su una delle macchine in modo che vengano connessi tra di loro, producento l'output finale.

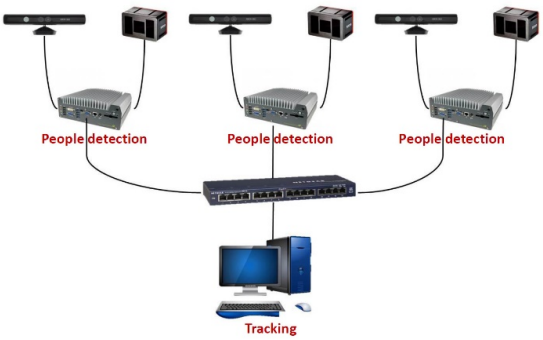


Figura 4.4: Schema di connessione tra computer e sensori per il funzionamento di OpenPTrack

Il punto forte di OpenPtrack risulta la capacità di funzionamento anche in ambienti chiusi con molti oggetti e condizioni di luce particolari.

Il funzionamento più nello specifico può essere riassunto nella seguente immagine: (Fig. 4.5)

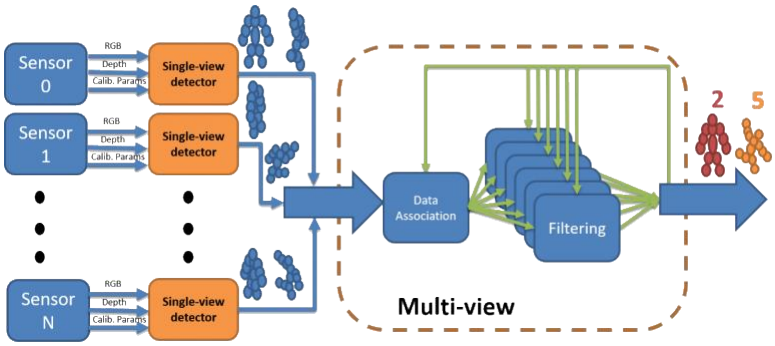


Figura 4.5: Funzionamento di OPT

Quindi ciascun computer, su cui è in esecuzione l'agoritmo, collegato ad una telecamera produce un'insieme di punti che sono la stima della posizione nello spazio 3D. Tale stima come indicato nella Figura 4.6 viene prodotta grazie ai dati 2D, i dati di profondità e i dati di calibrazione, che arrivano dai sensori collegati. Tutti questi dati vengono poi inviati al processo master, che restituisce in output la posizione definitiva nello spazio. Il calcolo della posa (scheletro) della persona per ogni telecamera è eseguito all'interno del singolo nodo tramite OpenPose, un algoritmo open source.

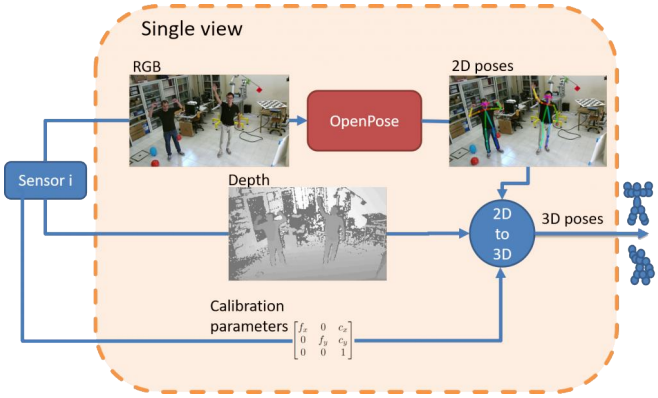


Figura 4.6: Schema del funzionamento di un singolo nodo

4.3 Sensori inerziali Xsens

Xsens è una parte dell'azienda Movella, nata dall'unione di Xsens, mCube e Kinduct. Questa parte dell'azienda si occupa della produzione di hardware e software per il tracciamento umano.

Tra i vari kit prodotti dall'azienda viene posto il focus sul kit MVN Awinda, composto da 17 sensori MTw Awinda e una stazione Awinda da connettere tramite USB al computer. Tutti i sensori sono dotati di accelerometri lineari 3D, giroscopi, magnetometri e barometri.

Da Xsens vengono messi a disposizione diversi software. Tra questi compare MVN Analyze, che viene utilizzato per raccogliere i dati provenienti dai sensori tramite la stazione Awinda. A partire dai dati grezzi il programma è in grando di ricostruire lo scheletro umano nello spazio. Nella Figura 4.7 vengono riportati lo scheletro ricostruito dal software e, in arancione, i sensori posizionati nel modo indicato dal costruttore in [57]. Il programma mette poi a disposizione la possibilità di ricevere, tramite streaming UDP, alcuni dati in vari formati elencati nell'user manual [59]. Tra i vari formati si ha la possibilità di ricevere le posizioni nello spazio di 23 giunti (x, y, z) con il loro orientamento sotto forma di quaternione (x, y, z, w). Dal costruttore vengono dichiarate, in base al numero di sensori attivi, le seguenti frequenze di trasmissione: 120 Hz con 1–5 MTw , 100Hz con 6–9 MTw, 80Hz con 10 MTw e 60Hz con 11–20 MTw.

La comunicazione tra i sensori MTw e la stazione avviene attraverso connessione wireless tramite il protocollo di comunicazione Awinda. Si tratta di un protocollo capace di rilevare e gestire perdite di pacchetti. La stazione manda, infatti, un segnale ad ogni sensore ogni volta che un pacchetto viene ricevuto. Se il sensore non riceve il segnale di feedback per la ricezione allora inserisce il

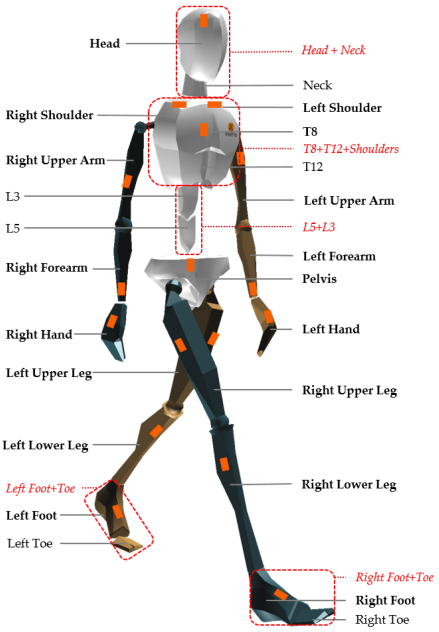


Figura 4.7: Ricostruzione scheletro con sensori xsens [58]

pacchetto in un buffer per la ritrasmissione.

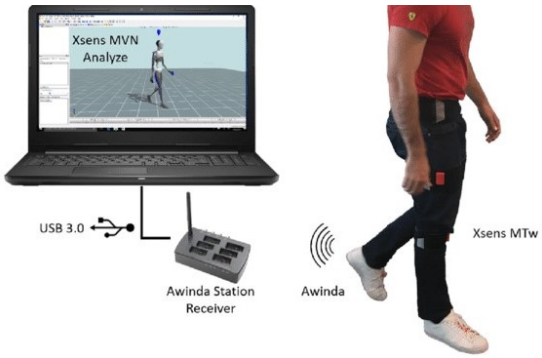


Figura 4.8: Connessioni tra gli elementi di Xsens [60]

4.4 Edge Impulse

Edge Impulse è un tool accessibile via internet che consente di sviluppare reti neurali utilizzabili su edge device. Con edge device s'intendono dispositivi come cellulari, laptop e in generale dispositivi embedded che possono eseguire elaborazioni locali sui dati senza la necessità di trasmetterli a un server remoto. Lo scopo dell'edge machine learning è quello di spostare l'esecuzione degli algoritmi di ML su dispositivi che si trovino in vicinanza ai sensori. In questo modo si può evitare la trasmissione di grandi quantità di dati attraverso la rete.

Nello specifico una delle parti che richiede maggiori risorse e potenza di calcolo è l'allenamento dei modelli di ML. Motivo per cui viene affidata ai server. Mentre la fase di inferenza, una volta prodotto il modello, non richiede grandi capacità di calcolo. Di conseguenza può essere affidata a dispositivi edge, dando vita al tinyML. In tal modo si ottiene una latenza minore, riducendo anche il consumo energetico.

Edge Impulse offre la possibilità di registrare i dati direttamente dai dispositivi edge (la piattaforma supporta moltissime tipologie) in modo da creare un dataset che possa essere poi utilizzato per la creazione dell'Impulso. Con il termine Impulso la piattaforma intende la composizione di un blocco di preprocessing e di un classificatore. Oggetto che dopo i test può essere caricato sul dispositivo edge, come indicato in Figura 4.9 dalla fase di Deploy. Va evidenziato che la registrazione dal parte del dispositivo può continuare anche quando l'Impulso è in esecuzione, consentendo così di aumentare la dimensione del dataset su cui si potra successivamente riallenare il modello. Da questo aspetto nasce il loop di Edge Impulse, riportato in Figura 4.9.

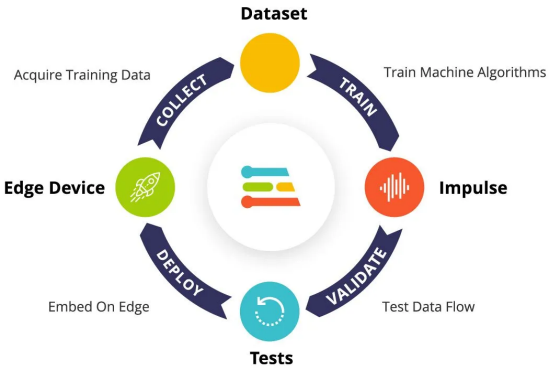


Figura 4.9: Il loop di Edge Impulse [61]

Un'alternativa all'acquisizione dei dati direttamente dal dispositivo edge è quella di caricare i file in formati adatti al training come csv, json, cbor, wav, jpg e png.

4.4.1 Struttura dell'impulso

Un impulso è composto da:

- Dati d'ingresso: possono essere una serie temporale o un'immagine
- Blocco di elaborazione: consente di effettuare il pre-processing

- Blocco di apprendimento: semplicemente una rete neurale addestrata per apprendere dai dati
- Output features: i dati in uscita, ad esempio i possibili label

Nella fase di creazione dell'impulso, per il confronto sperimentale, è stato prodotto un impulso con dati d'ingresso sotto forma di serie temporale, salvati in un file csv. Il blocco di elaborazione che effettua il preprocessing utilizza la spectral analysis. Mentre in fase di apprendimento sono state sperimentate molteplici configurazioni di reti neurali. I parametri e gli strumenti vengono approfonditi in seguito.

Capitolo 5

Esperimenti

La parte sperimentale del lavoro si è basata sull'utilizzo dello strumento ROS per creare un dataset e registrare i dati relativi ai giunti provenienti dalle varie piattaforme e salvarli in dei file. Lo scopo principale di poter riprodurre le registrazioni è stato facilitare lo sviluppo e, soprattutto, consentire la conversione in un formato ammissibile al training di una rete neurale. Nello specifico, si è fatto uso del formato CSV, molto popolare per il traning di reti neurali e accettato da Edge Impulse. Dopo aver effettuato la conversione è stato possibile effettuare l'allenamento di più reti neurali per la classificazione delle azioni umane, con l'obiettivo di trovare la migliore.

Tutti gli esperimenti sono stati effettuati presso l'Università di Padova, nell'IAS-Lab del Dipartimento di Ingegneria dell'informazione [62].

Una struttura basilare degli strumenti utilizzati e di come siano collegati tra di loro viene riportata nella Figura 5.1 .

5.1 Creazione del dataset

Il primo compito da svolgere è stato quello di raccogliere un dataset su cui effettuare il training e con cui verificare le performance ottenute. Le azioni scelte per il dataset sono: immobile, camminare in avanti, squat, raccogliere una bottiglia, nuotare a stile, camminare indietro, fare delle flessioni e saltare. Tale insieme di azioni è stato scelto per rappresentare vari movimenti distintivi che può compiere una persona, enfatizzando la presenza di azioni simili tra loro (camminare avanti, camminare indietro). Considerando uno scenario di collaborazione uomo-robot come quello scelto a motivazione della tesi, si possono associare la maggior parte delle azioni scelte ad una precisa situazione.

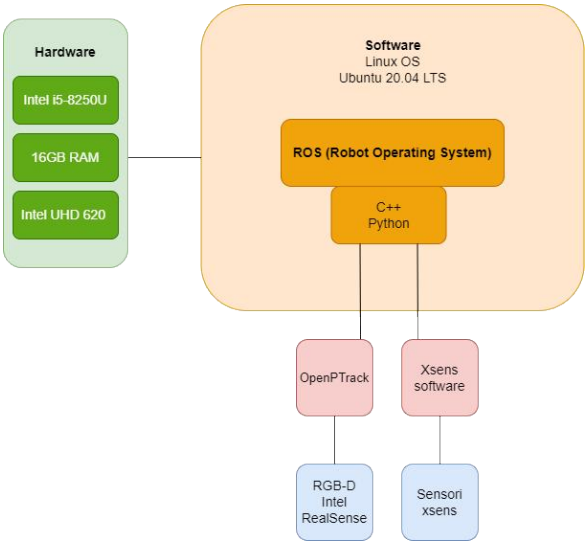


Figura 5.1: struttura software usata per gli esperimenti

L'immobilità in una collaborazione uomo-robot potrebbe rappresentare un momento in cui il robot si fermi per consentire all'essere umano di eseguire un'azione specifica o di prendere decisioni. Mentre camminare in avanti potrebbe rappresentare il robot che segue l'essere umano, magari in un contesto di assistenza o supporto. Il robot potrebbe spostarsi in modo sincronizzato all'andatura dell'essere umano per aiutarlo nei suoi movimenti.

I movimenti di squat possono essere collegati alla collaborazione uomo-robot nel contesto di un sollevamento o di un lavoro che richiede una forza fisica maggiore. Il robot potrebbe sostenere l'essere umano durante i movimenti di squat riducendo il carico per l'essere umano.

Mentre per la raccolta di una bottiglia, l'azione può essere collegata alla collaborazione uomo-robot nel contesto di una mansioni di pulizia o di organizzazione del luogo di lavoro. Il robot potrebbe essere programmato per fermarsi durante il processo di pulizia o eseguire operazioni che facilitino il lavoro dell'operatore.

L'azione di camminare all'indietro da parte dell'uomo potrebbe rappresentare una situazione in cui il robot segue l'essere umano mentre quest'ultimo guida i movimenti attraverso un ambiente angusto.

Se l'uomo esegue l'azione di saltare, il robot potrebbe essere coinvolto nella collaborazione fornendo supporto o assistenza durante il salto. Ad esempio, potrebbe stabilizzare l'essere umano durante il salto. O determinare che il salto dell'utente implichi una situazione di pericolo, bloccando azioni che possano risultare un rischio per la sicurezza.

Sono poi state aggiunte due situazioni facilmente distinguibili, ovvero nuotare

e fare flessioni, in modo da poter verificare le capacità di riconoscimento su azioni più semplici.

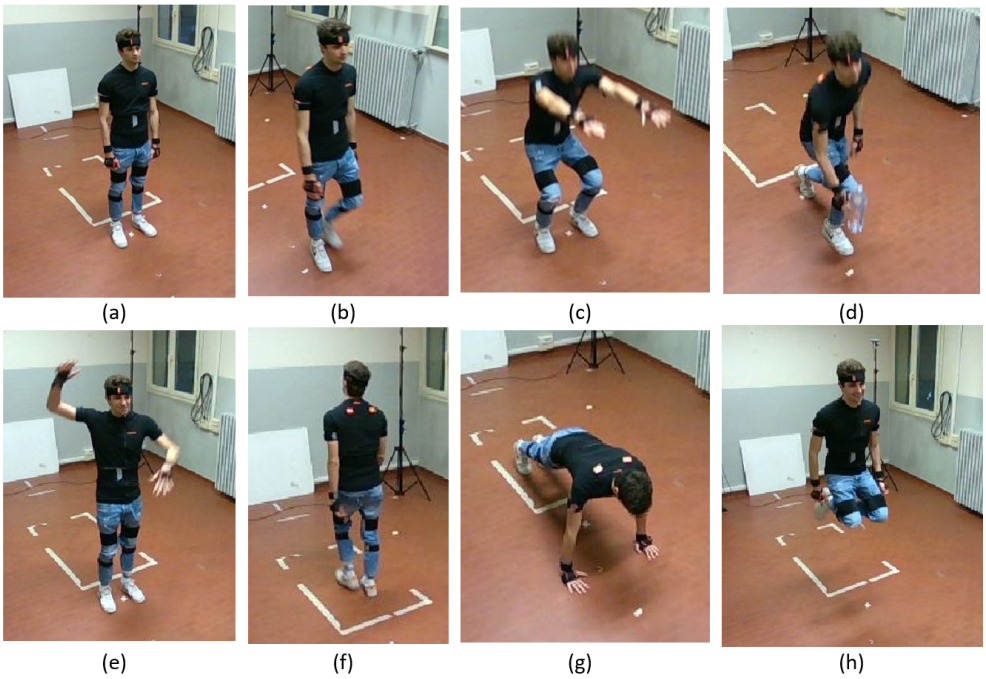


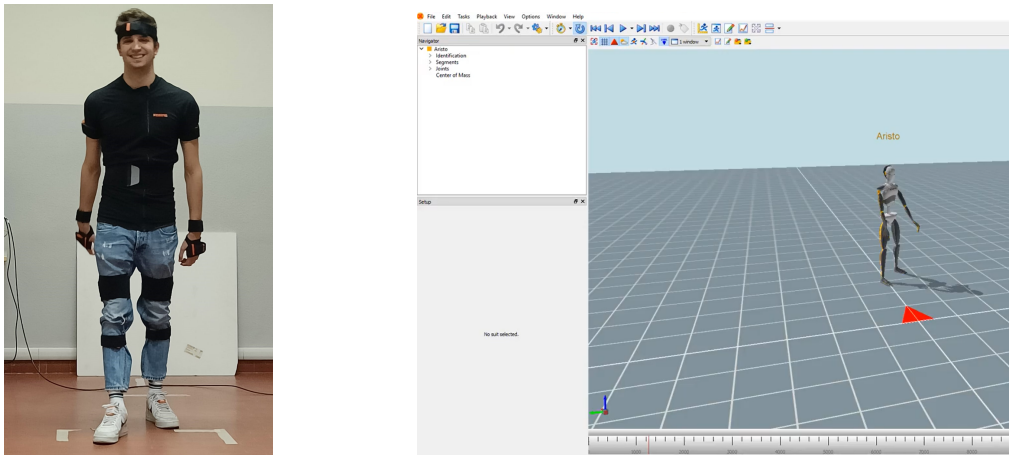
Figura 5.2: Frame presi dal dataset corrispondenti alle seguenti azioni: (a) immobile, (b) camminare in avanti, (c) squat, (d) raccogliere una bottiglia, (e) nuotare a stile, (f) camminare indietro, (g) fare delle flessioni e (h) saltare.

Ciascuna azione è stata riprodotta da 3 persone diverse di sesso maschile, in modo da introdurre delle differenze di altezza e dimensioni anatomiche, per 5 volte. Producendo così un dataset di $8 * 3 * 5 = 120$ campioni.

5.1.1 IMU

Per quanto concerne i sensori inerziali (IMU) sono stati utilizzati 3 kit MTw2-DK-6 S/N, prodotti da Xsens. Ciò perché ciascun kit contiene 6 sensori. Negli esperimenti i sensori MTw attivi, 17 necessari per la ricostruzione della posa dell'intero corpo, sono stati posizionati nel modo consigliato da Xsens in [57].

Al fine di ottenere una corretta ricostruzione del corpo umano è necessaria una calibrazione iniziale dei sensori, in cui è necessario fornire misurare parti del corpo come altezza e lunghezza del piede. Viene poi richiesto di stare fermi in T pose, ovvero con le braccia distese orizzontalmente ai lati del corpo, e N pose, cioè con le braccia verticalmente ai lati del corpo, per qualche secondo.



(a) (b)

Figura 5.3: Nell'immagine (a) il posizionamento dei sensori, nell'immagine (b) una schermata del software MVN Analyze

Il software MVN Analyze (Figura 5.3) di Xsens offre più modalità di streaming dei dati in modalità UDP. In questo esperimento è stata utilizzata la modalità che offre la posizione assoluta nello spazio dei 23 giunti come un vettore di coordinate 3D (x, y, z) e l'orientamento come una quaternione (x, y, z, w). Si vuole far notare che la posizione è assoluta rispetto ad un punto di partenza. Tale punto viene definito al momento della calibrazione oppure ridefinito manualmente tramite un comando messo a disposizione da MVN Analyze.

I dati dello streaming UDP tramite un nodo ROS già sviluppato dall'IAS-Lab UniPD vengono pubblicati come messaggi ROS e registrati in dei file formato .bag, in modo da poterne garantire la riproduzione successivamente.

5.1.2 RGB-D

Per la raccolta dei dati tramite telecamere RGB-D sono state utilizzate 4 videocamere Intel RealSense D455 [63] posizionate a circa 2 metri da terra. In Figura 5.4 viene riportato il setup utilizzato nella fase di raccolta del dataset. Nell'immagine si può notare la griglia utilizzata nella calibrazione della rete di telecamere per calcolare la loro posizione reciproca.

Ciascuna delle videocamere è stata collegata ad un computer con la distro Ubuntu di Linux. I 4 computer sono stati poi collegati via Gigabit Ethernet ad un computer Master con installato OpenPTrack v2 Gnocchi. Il software ha consentito di produrre in tempo reale uno scheletro tridimensionale. Essendo basato su ROS il programma di OPT pubblica anche le posizioni dei giunti come dei

Figura 5.4: Setup delle videocamere Intel RealSense D455 utilizzate per la raccolta del dataset.

messaggi su un topic. I messaggi utilizzati sono disponibili sulla pagina GitHub del progetto [64]. E' stato quindi possibile registrare i topic corretti su dei file .bag, in modo da poter effettuare la riproduzione.

5.1.3 Problematiche

Durante la fase di acquisizione del dataset tramite i sensori MTw di Xsens si è verificato un problema con la frequenza di arrivo dei dati da parte dei sensori.

Le frequenze di registrazione, nonostante la capacità di arrivare fino a 60Hz visto il numero di sensori utilizzati, sono rimaste a 50Hz effettuando la registrazione su computer con le seguenti caratteristiche:

- Windows 10 (64 bit)
- Intel(R) Core(TM) i5-8250U CPU (Quad core con velocità di clock massima 3.4 GHz)
- SSD 512GB
- 16GB di RAM

Oltre alla diminuzione della frequenza, il processo di calibrazione dei sensori inerziali è diventato particolarmente lento. Queste problematiche si sono presentate nonostante le richieste minime riportate da Xsens per quanto riguarda i software MVN siano:

- Windows 10 (64 bit)

- Quad core or higher (2.7 GHz or faster)
- SSD 256 GB or more
- 8 GB of RAM

E' importante evidenziare che con macchine di potenza superiore a quelle utilizzate da noi la frequenza riscontrata risultava effettivamente 60 Hz.

5.2 Conversione da file bag a file csv

I file bag non vengono accettati da Edge Impulse. Motivo per cui sono stati sviluppati due programmi analoghi che prendono i file all'interno di una cartella indicata dall'utente e li convertono da file bag a file csv. Fondamentale in questo caso è la conversione effettuata da ROS dei messaggi in classi, potendo così accedervi in modo semplice come se si trattasse di oggetti.

I programmi sviluppati sono stati scritti entrambi in C++, utilizzando le librerie messe a disposizione da ROS. Nell'appendice 6.2 viene presentato il programma sviluppato per effettuare la conversione nel caso degli scheletri ottenuti da OpenPTrack. Si tralascia il programma per la conversione dei dati ottenuti dai sensori Xsens in quanto analogo.

Nel codice si può notare `std::vector<int> skeleton_used`. Un vettore contente dei numeri, che possono sembrare casuali. Il vettore si è reso necessario poiché le videocamere inquadravano anche altre persone e, inoltre, il soggetto usciva spesso dall'inquadratura. OPT va ad assegnare un numero progressivo ad ogni scheletro che entra nell'inquadratura da fuori. Motivo per cui ha assegnato più valori, che sono stati inseriti nel vettore, alla stessa persona. Nel codice viene quindi utilizzato uno scheletro con un certo ID finché disponibile, appena questo non è più presente si ricerca tra gli scheletri presenti nell'istante di tempo un elemento con ID che compaia all'interno del vettore.

5.3 Sviluppo della rete neurale

Come già detto nel Capitolo 3 la rete neurale sviluppata si basa sul software Edge impulse, che si basa sulla creazione di un impulso composto da 4 parti: dati d'ingresso, blocco di elaborazione, blocco di apprendimento e output features.

5.3.1 Dati d'ingresso

Al momento della selezione vengono chiesti i seguenti parametri:

- Window size: si può decidere, quando si hanno delle ripetizioni continue di dividere la scena in frame di una determinata lunghezza
- Window increse: determina di quanti sample scorrere prima di avere un nuovo campionamento delle dimensioni di window size. Se si pone window increse pari al window size significa che non ci saranno frame con overlapping
- Frequency: viene calcolata in automatico a partire dai sample
- Zero-pad data: aggiunge degli zeri nel caso il window size sia maggiore della lunghezza dei dati

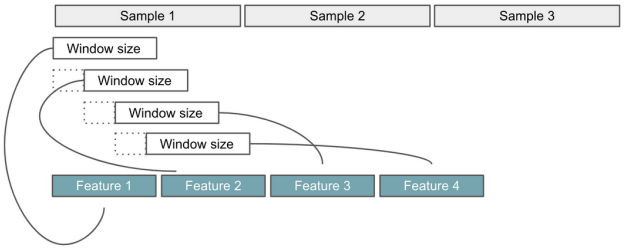


Figura 5.5: Spiegazione di window size e window increse [61]

Per via della natura ripetitiva dei movimenti utilizzati e della dimensione ridotta del dataset è stato preferibile utilizzare dei Window size minori della lunghezza delle registrazioni. In questo modo vengono estratte delle feature per ogni frame della dimensione di window size. Sono stati quindi utilizzati dei window size e dei window increase entrambi di 1000ms.

La frequenza di entrambi i dataset è stata acquisita in modo corretto.

Inoltre, tramite le varie impostazioni di Edge impulse si otterrà una suddivisione dei campioni di input in 3 categorie:

- Training set: input utilizzati dall'algoritmo di training per determinare le modifiche da fare ai parametri interni, come pesi e bias.
- Validation set: input utilizzati dal modello per determinare quali parametri, come epoche e learning rate, portano alle prestazioni migliori per la rete.

- Test set: input utilizzati per verificare le prestazioni della rete finale su dati mai visti, ha lo scopo fondamentale di verificare se si è incorsi nel fenomeno dell'overfitting (ovvero la rete allenata ha ottime prestazioni sui dati di allenamento, ma non è in grado di generalizzare su nuovi dati).

5.3.2 Pre-processing

Successivamente viene data la possibilità di introdurre un pre-processing del segnale in modo da rimuovere il rumore. Nello specifico in questo lavoro è stata utilizzata la spectral analysis di Edge Impulse. Quest'ultima va ad estrarre frequenze e caratteristiche tipiche di un segnale, applicando eventualmente filtri passa alto e filtri passa basso. Risulta particolarmente utile per quanto riguarda l'analisi di pattern ripetitivi in segnali come movimenti, vibrazioni o vibrazioni da accelerometri. Nel progetto non sono stati utilizzati filtri particolari, mentre per la fase di analisi sono stati utilizzati specifici parametri per quanto concerne le trasformate di Fourier. La trasformata di Fourier è impossibile da implementare all'interno di un computer poiché si tratta di un integrale che richiederebbe un intervallo infinito. Inoltre in un computer è impossibile rappresentare un infinitesimo. Quindi si discretizza l'integrale e si mettono degli estremi di integrazione finiti. Facendo questo si ha la DFT, che è la serie di Fourier a tempo discreto. FFT è un algoritmo che calcola in modo efficiente la DFT [65].

Abbiamo quindi utilizzato FFT, di cui sono stati settati determinati parametri. Il più importante è il fatto di porre a 16 il numero di sample per ogni finestra che viene utilizzata (parametro chiamato FFT length in Edge Impulse), dato che per avere un algoritmo di FFT veloce tale parametro deve essere multiplo di 2. Inoltre, è stata abilitata la capacità di avere un overlapping di 1/2 per quanto concerne i frame in cui viene calcolata la FFT. Per maggiori dettagli si rimanda alla documentazione di Edge Impulse [61].

L'obiettivo di questa fase è di considerare le coordinate dei vari giunti come dei segnali, a cui applicare la trasformata di Fourier in modo veloce. Tramite la FTT si è così potuta mettere in evidenza la potenza delle frequenze dominanti presenti nel segnale. Ciò si basa sul concetto che, come evidenziato in [1], generalmente le persone hanno un modo simile di effettuare determinate azioni.

5.3.3 Classificazione

Nella fase di classificazione viene richiesto di impostare i seguenti parametri:

- Number of training cycles: detti anche epoch, indicano il numero di volte che l'algoritmo di allenamento fa un passaggio completo attraverso tutti i dati di training e tramite la back-propagation va ad aggiornare i parametri del modello.

- Learning rate: controlla quanto i parametri interni sono aggiornati durante ogni passo del processo di training. Può essere anche visto come la velocità con cui il modello impara.

- Validation set size: la percentuale del training set che viene tenuto da parte per il processo di validazione.

- Auto-balance: sopperisce al basso numero di dati per alcune classi ripresentandole più spesso, può aiutare a combattere l'overfitting.

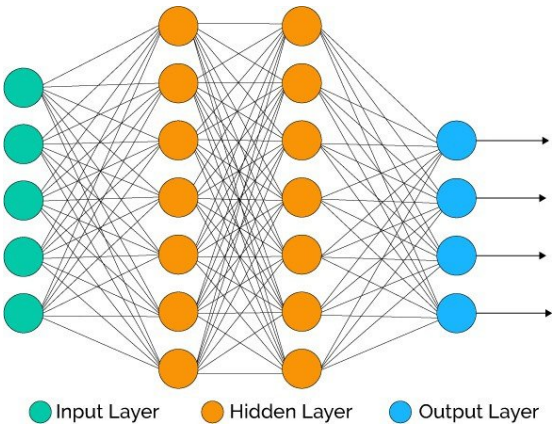


Figura 5.6: Schema simile alle reti neurali analizzate

Come validation set size si è scelta una partizione dei dati di training che implica un 20% di validazione.

Per questo lavoro è stata utilizzata una rete neurale di tipo Multilayer Perceptron (MLP) in cui il numero di neuroni nell'input layer, ovvero il numero di feature in ingresso, è stato fissato sulla base del tipo di sensore utilizzato. Per i dati ricavati dalle videocamere RGB-D si ha un input di 495 feature, mentre per i dati ottenuti dai sensori innerziali si arriva a 1771 feature in ingresso. Anche il numero di neuroni per l'output layer è stato fissato a 8 in entrambi i casi.

Mentre per il learning rate, i training cycles, il numero di hidden layer e il numero di neuroni all'interno degli hidden layer si è deciso di sperimentare più possibilità.

5.3.4 Ricerca modello migliore

Allo scopo di trovare la rete che desse le prestazioni migliori sono state provate diverse configurazioni. Si riportano le Accuracy ottenute sul validation set (VALI.), che sul test set (TEST). Il test set è il 20% dei campioni raccolti, e mentre il validation set per determinare i pesi e i bias è il 20% dei campioni non appartenenti al test set. I risultati ottenuti con varie reti e utilizzando un tasso di apprendimento (LR) pari a 0.0005 e 100 cicli di addestramento (TC) vengono riportati in Tabella 5.1. Mentre in Tabella 5.2 vengono presentati i risultati prodotti dalle reti con tasso di apprendimento 0.0005 e 200 cicli di addestramento. Infine, in tabella 5.3 sono esposti i valori prodotti da un tasso di apprendimento 0.0001 e 200 cicli di addestramento.

TC=100, LR=0.0005				IMU		RGB-D	
NEUR.	LAYER1	LAYER2	LAYER3	VALI.	TEST	VALI.	TEST
3	40	20	10	92.1%	83.16%	79.5%	74.19%
3	100	90	30	89.5%	87.37%	86.3%	80.65%
3	40	30	20	88.2%	87.37%	86.3%	79.57%
2	30	10	no	84.2%	78.95%	80.8%	73.12%
2	100	90	no	89.5%	90.53%	83.6%	81.72%
2	40	30	no	92.1%	82.11%	82.2%	76.34%
2	120	40	no	85.5%	84.21%	85.5%	75.27%
2	50	100	no	88.2%	89.47%	82.2%	76.34%

Tabella 5.1: Risultati ottenuti con tasso di apprendimento pari a 0.0005 e 100 cicli di apprendimento.

TC=200, LR=0.0001				IMU		RGB-D	
NEUR.	LAYER1	LAYER2	LAYER3	VALI.	TEST	VALI.	TEST
3	40	20	10	89.5%	87.37%	78.1%	64.52%
3	100	90	30	89.5%	91.58%	84.9%	70.97%
3	40	30	20	86.8%	85.26%	80.8%	69.89%
2	30	10	no	86.8%	75.79%	78.1%	65.59%
2	100	90	no	86.8%	85.26%	83.6%	81.72%
2	40	30	no	90.8%	83.16%	80.8%	74.19%
2	120	40	no	86.8%	85.26%	82.2%	80.65%
2	50	100	no	92.1%	88.42%	86.3%	80.65%

Tabella 5.2: Risultati ottenuti con tasso di apprendimento pari a 0.0001 e 200 cicli di apprendimento.

TC=200, LR=0.0005				IMU		RGB-D	
NEUR.	LAYER1	LAYER2	LAYER3	VALI.	TEST	VALI.	TEST
3	40	20	10	88.2%	86.32%	79.5%	74.19%
3	100	90	30	89.5%	91.58%	84.9%	70.97%
3	40	30	20	88.2%	87.37%	86.3%	79.57%
2	30	10	no	82.9%	87.37%	80.8%	73.12%
2	100	90	no	89.5%	90.53%	83.6%	81.72%
2	40	30	no	92.1%	82.11%	82.2%	76.34%
2	120	40	no	85.5%	84.21%	83.6%	75.27%
2	50	100	no	90.8%	91.58%	82.2%	76.34%

Tabella 5.3: Risultati ottenuti con tasso di apprendimento pari a 0.0005 e 200 cicli di apprendimento.

Sono stati effettuati anche test su altre configurazioni in cui si è tenuto fisso il numero di Training Cycles a 100, il learning rate a 0.0005 e il numero di layer a 2. Sono invece stati fatti variare i numeri di neuroni che compongono i due layer, come riportato in Tabella 5.4.

TC=100, LR=0.0005			IMU		RGB-D	
NEUR.	LAYER1	LAYER2	VALI.	TEST	VALI.	TEST
2	10	30	84.2%	76.84%	80.8%	73.12%
2	10	20	76.7%	69.89%	78.9%	56.84%
2	10	10	68.5%	44.09%	10.5%	0.00%
2	10	40	53.4%	27.96%	27.6%	1.05%
2	30	40	84.9%	75.27%	64.5%	31.58%
2	50	120	88.2%	88.42%	80.8%	73.12%
2	90	150	89.5%	87.37%	83.6%	67.74%
2	10	30	10.5%	0.00%	54.8%	38.71%

Tabella 5.4: Ulteriori risultati ottenuti con tasso di apprendimento pari a 0.0005 e 100 cicli di apprendimento.

Per i sensori inerziali, quindi, il risultato migliore è stato ottenuto con un Multilayer Perceptron contenente due hidden layer: il primo costuito da 50 neuroni e il secondo da 100. Con un tasso di apprendimento pari a 0.0005 e 200 cicli di addestramento si è raggiunta un Accuracy sul set di validazione del 90.8% e del 91.58% sul set di test.

Mentre, per i sensori RGB-D la rete che si è dimostrata più efficacie è un Multilayer Perceptron con due hidden layer: il primo di 100 neuroni e il secondo di 90. Il tasso di apprendimento che ha mostrato risultati ottimi è 0.0001, men-

tre i cicli di addestramento migliori sono 200. L'Accuracy raggiunta da questa configurazione è 83.6% sul set di validazione e 81.72% sul set di test.

5.3.5 Impulso ottenuto

L'impulso finale per i dati ottenuti tramite IMU è riportato in Figura 5.7.

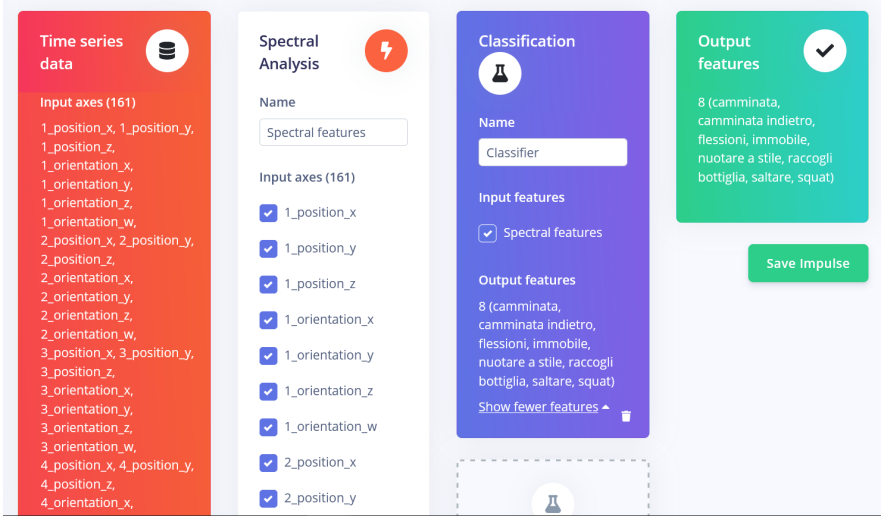


Figura 5.7: Impulso finale

Si evita di riportare l'impulso per le immagini RGB-D in quanto analogo, con l'unica differenza di avere meno input axes.

Nelle figure 5.8 e 5.9 vengono riportati, sottoforma di confusion matrix, i risultati ottenuti sul validation set in termini di Accuracy, percentuale di errore e in ultima riga F1 Score. Con F1 Score s'intende il valore ottenuto dalla seguente formula:

$$F1score = 2 * \frac{precision * recall}{precision + recall}$$

Precision indica le volte in cui l'aver predetto una certa classe si è dimostrato corretto. Mentre Recall indica quanto spesso una classe viene predetta correttamente. L'Accuracy è una metrica per valutare i modelli di classificazione, definibile come segue:

$$Accuracy = \frac{\text{Numero di predizioni corrette relative a sample x}}{\text{Numero totale di predizioni relative al sample x}}$$

La percentuale di errore è definibile come:

$$\text{Perc. errore(x,y)} = \frac{\text{Label y quando sample x}}{\text{Numero totale di predizioni relative al sample x}}$$

con $x \neq y$.

Nelle figure 5.8, 5.9, 5.10 e 5.11 i label associati ai valori sono: 1) immobile; 2) camminare in avanti; 3) squat; 4) raccogliere bottiglia; 5) nuotare a stile; 6) camminare indietro; 7) fare delle flessioni; 8) saltare.

Figura 5.8: Risultati ottenuti sul validation set per i dati prodotti tramite IMU.

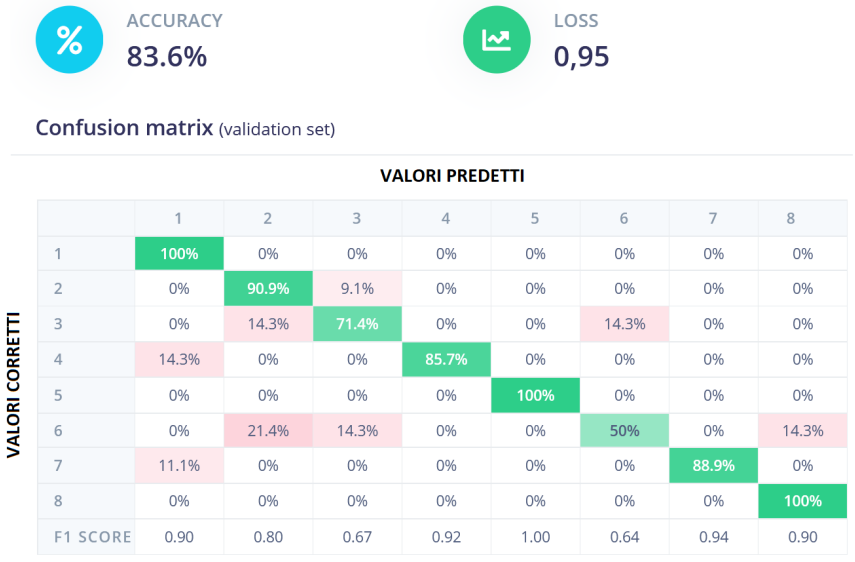


Figura 5.9: Risultati ottenuti sul validation set per i dati prodotti tramite videocamere RGB-D.

Sulla diagonale principale, in verde, si trovano le percentuali di Accuracy precedentemente definita. Per ciascuna riga si ha la percentuale di volte in cui un sample con un certo label corretto viene predetto con il label relativo alla colonna.

5.3.6 Model testing

L'impulso può finalmente essere testato sul set di test. In Figura 5.11 viene mostrata la confusion matrix ottenuta tramite l'impulso generato dai dati RGB-D, mentre in Figura 5.10 la confusion matrix prodotta dall'impulso derivato da IMU.

Figura 5.10: Risultati ottenuti sul test set per i dati prodotti tramite IMU.

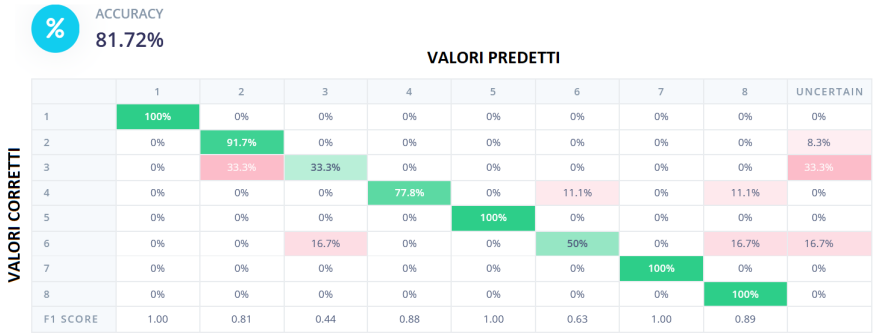


Figura 5.11: Risultati ottenuti sul test set per i dati prodotti tramite videocamere RGB-D.

5.3.7 Problematiche

Durante il primo caricamento i dati non sono stati accettati da Edge Impulse. Questo per due motivi principali: alcuni dati risultavano avere lo stesso time-

stamp e l'intervallo tra i vari sample risultava non costante. Cosa necessariamente richiesta dalla piattaforma, come sottolineato nella documentazione. Tramite questi errori si è scoperto, dopo un'analisi, che l'ordine dei sample veniva rispettato ma i timestamp di ROS (introdotti in automatico nel header del messaggio) risultavano avere intervalli non costanti tra di loro. Questa problematica di ROS viene approfondita più accuratamente da diversi autori, tra cui [66].

Per sopperire al problema, visto che viene rispettato comunque l'ordine dei campioni, è stata inserita una riga nel codice `(timestamp*x)` con `x` pari all'intervallo tra i vari sample. Inoltre, grazie a questa scoperta è stato poi modificato il codice utilizzato dall'IAS-Lab in modo da trasmettere tra i vari dati anche il timestamp relativo ai sensori, in quanto corretto.

Capitolo 6

Conclusioni e sviluppi futuri

6.1 Conclusioni

Questo lavoro non vuole essere un punto di arrivo, ma piuttosto un punto d'inizio per analizzare le differenze tra le due tecnologie di estrazione dello scheletro umano nel contesto del riconoscimento di azioni umane.

L'approccio di utilizzare per il riconoscimento i giunti del corpo umano si è dimostrato efficacie e porta con se un grosso vantaggio in termini di velocità nella fase di training e di riconoscimento. Modalità come l'utilizzo di immagini hanno, infatti, lo svantaggio di rallentare particolarmente queste funzioni visto che la velocità dipende dal numero di sample, ma soprattutto dalla loro dimensione.

Mentre per quanto riguarda l'utilizzo dei giunti come alternativa ai raw data dei sensori inerziali è stato verificato, tramite opportuni articoli, che si introduce così un invarianza a traslazioni e rotazioni. Robustezza non presente senza il passaggio attraverso lo scheletro.

Successivamente, è stato possibile verificare che i dati ottenuti da IMU portano ad Accuracy maggiore, come supposto in principio. Infatti, nel model testing si è ottenuta un'Accuracy del 91.58% con gli IMU e del 81.72% utilizzando le videocamere RGB-D. I motivi principale risultano essere due: la precisione dei dati raccolti, maggiore per gli IMU, e la frequenza con cui vengono prodotti i dati. Infatti, OPT produce i dati con frequenza di 30Hz, mentre gli IMU inviano i dati con frequenza di 50Hz. Inoltre, i sensori oltre a presentare le posizioni dei giunti (che sono 23 contro i 16 degli scheletri estratti da immagini) presentano anche l'orientamento delle giunzioni. Ragione per cui la maggiore precisione ottenuta con le unità di misura inerziali è in linea con le ipotesi. Si può quindi affermare il dominio di questa tecnica, nonostante porti con se diversi svantaggi e vantaggi

sotto altri termini già descritti in precedenza.

L'esperimento ha portato ad una qualità dei risultati buona considerando il basso numero di registrazioni, se confrontato con le dimensioni dei dataset della letteratura. Anche se sicuramente l'utilizzo di alcune azioni particolarmente diverse tra di loro ha influito positivamente nella riuscita.

Una critica che può essere mossa a questo confronto è che la bontà dei risultati sia legata all'utilizzo di software come OPT e Xsens. Si tratta tuttavia di alcuni dei software più utilizzati quindi è possibile ignorare le differenze in termini di precisione da parte dei due.

Inoltre, bisogna evidenziare che la maggior precisione da parte dei sensori inerziali non significa necessariamente che siano l'approccio corretto da utilizzare in tutti i contesti. Entrambi gli approcci portano con se determinati vantaggi e svantaggi. Risulta quindi fondamentale un'analisi in base al contesto di utilizzo per una corretta scelta della tipologia di dispositivi da utilizzare.

6.2 Sviluppi futuri

Il lavoro offre la possibilità di procedere con la raccolta di azioni più affini alla collaborazione uomo-robot e altre applicazioni più specifiche. Esempi di azioni relative al contesto HRC potrebbero essere quelle utilizzate in industrie come la già citata Drapebot [5] dove si ha la stesura di materiale su degli stampi. Oppure la situazione in cui si stringe una vite, ecc.

Un'ulteriore sviluppo sarebbe effettuare lo stesso confronto, ma utilizzando tecniche di classificazione più avanzate come quelle riportate nel Capitolo 2. Inoltre, espandendo la classificazione ad un dataset di dimensioni maggiori. In modo da poter verificare la differenza su azioni più specifiche.

Grazie all'utilizzo di Edge Impulse, le librerie prodotte durante questo lavoro possono essere applicate in uno scenario reale tramite i file scaricabili. E' quindi possibile implementare l'utilizzo di Edge impulse all'interno di un nodo ROS in modo da avere un riconoscimento delle azioni live, abbracciando a pieno la politica Edge.

Infine, sarebbe corretto effettuare ulteriori versioni degli esperimenti utilizzato le modalità cross subject, ovvero utilizzando soggetti diversi tra training e model testing, e producendo una quantità di registrazioni tali da poter effettuare un model testing sempre su campioni diversi, in modo da non perdere di generalizzazione.

Bibliografia

Capitolo 1

[1] Zehua Sun et al. «Human Action Recognition From Various Data Modalities: A Review». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), pp. 1–20 (cit. alle pp. 2, 19, 42).

[2] Matteo Munaro, Filippo Basso e Emanuele Menegatti. «OpenPTrack: Open source multi-camera calibration and people tracking for RGB-D camera networks». In: *Robotics and Autonomous Systems* 75 (2016), pp. 525–538. ISSN: 0921-8890 (cit. a p. 2).

[3] S. Robla-Gómez et al. «Working Together: A Review on Safe Human-Robot Collaboration in Industrial Environments». In: *IEEE Access* 5 (2017), pp. 26754–26773 (cit. a p. 3).

[4] Stefano Ghidoni et al. «A Smart Workcell for Human-Robot Cooperative Assembly of Carbon Fiber Parts». In: ott. 2021 (cit. alle pp. 3, 8).

[6] Qianqian Xiong et al. «Transferable two-stream convolutional neural network for human action recognition». In: *Journal of Manufacturing Systems* 56 (2020), pp. 605–614. ISSN: 0278-6125 (cit. alle pp. 4, 19, 20).

[9] Kyriaki A. Tychola, Ioannis Tsimperidis e George A. Papakostas. «On 3D Reconstruction Using RGB-D Cameras». In: *Digital* 2.3 (2022), pp. 401–421. ISSN: 2673-6470 (cit. a p. 5).

[10] Alexandre Lopes, Roberto Souza e Helio Pedrini. «A survey on RGB-D datasets». In: *Computer Vision and Image Understanding* 222 (2022), p. 103489. ISSN: 1077-3142 (cit. a p. 5).

[49] J.K. Aggarwal e M.S. Ryoo. «Human Activity Analysis: A Review». In: *ACM Comput. Surv.* 43.3 (apr. 2011). ISSN: 0360-0300 (cit. a p. 18).

Capitolo 2

[11] Nicolas Valencia-Jimenez. «A Comparative Study of Markerless Systems Based on Color-Depth Cameras, Polymer Optical Fiber Curvature Sensors, and Inertial Measurement Units: Towards Increasing the Accuracy in Joint Angle Estimation». In: *Electronics* (2019) (cit. a p. 11).

[12] Halim Tannous et al. «A New Multi-Sensor Fusion Scheme to Improve the Accuracy of Knee Flexion Kinematics for Functional Rehabilitation Movements». In: *Sensors* 16.11 (2016). ISSN: 1424-8220 (cit. a p. 11).

[13] Luis Roda-Sanchez et al. «Comparison of RGB-D and IMU-based gesture recognition for human-robot interaction in remanufacturing». In: *The International Journal of Advanced Manufacturing Technology* 124.9 (feb. 2023), pp. 3099–3111 (cit. a p. 11).

[14] Ong Chin Ann e Lau Bee Theng. «Human activity recognition: A review». In: *2014 IEEE International Conference on Control System, Computing and Engineering (ICCSCE 2014)*. 2014, pp. 389–393 (cit. a p. 12).

[16] Songyang Zhang, Xiaoming Liu e Jun Xiao. «On Geometric Features for Skeleton-Based Action Recognition Using Multilayer LSTM Networks». In: mar. 2017 (cit. a p. 13).

[17] Jun Liu et al. «Skeleton-Based Human Action Recognition With Global Context-Aware Attention LSTM Networks». In: *IEEE Transactions on Image Processing* 27.4 (2018), pp. 1586–1599 (cit. a p. 13).

[18] Rasha Friji et al. «Geometric Deep Neural Network using Rigid and Non-Rigid Transformations for Human Action Recognition». In: nov. 2021 (cit. alle pp. 13, 14).

[19] Yong Du, Wei Wang e Liang Wang. «Hierarchical recurrent neural network for skeleton based action recognition». In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1110–1118 (cit. a p. 13).

[20] Rikiya Yamashita et al. «Convolutional neural networks: an overview and application in radiology». In: *Insights into Imaging* 9.4 (ago. 2018), pp. 611–629. ISSN: 1869-4101 (cit. a p. 13).

[21] Huy Hieu Pham et al. «Spatio–Temporal Image Representation of 3D Skeletal Movements for View-Invariant Action Recognition with Deep Convolutional Neural Networks». In: *Sensors* 19.8 (2019). ISSN: 1424-8220 (cit. a p. 13).

[22] Miao Feng e Jean Meunier. «Skeleton Graph-Neural-Network-Based Human Action Recognition: A Survey». In: *Sensors* 22.6 (2022). ISSN: 1424-8220 (cit. a p. 13).

[23] Xikun Zhang et al. «Graph Edge Convolutional Neural Networks for Skeleton-Based Action Recognition». In: *IEEE Transactions on Neural Networks and Learning Systems* PP (set. 2019), pp. 1–14 (cit. a p. 13).

[24] Matthew Korban e Xin Li. «DDGCN: A Dynamic Directed Graph Convolutional Network for Action Recognition». In: *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX*. Berlin, Heidelberg: Springer-Verlag, 2020, pp. 761–776. ISBN: 978-3-030-58564-8 (cit. a p. 13).

[25] Hyung-gun Chi et al. «InfoGCN: Representation Learning for Human Skeleton-Based Action Recognition». In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Giu. 2022, pp. 20186–20196 (cit. alle pp. 13, 14).

[27] Ashish Vaswani et al. *Attention Is All You Need*. 2017. arXiv: `1706.03762 [cs.CL]` (cit. a p. 14).

[28] Shuo Chen et al. «Pyramid Spatial-Temporal Graph Transformer for Skeleton-Based Action Recognition». In: *Applied Sciences* 12.18 (2022). ISSN: 2076-3417 (cit. a p. 14).

[29] Wentian Xin et al. «Transformer for Skeleton-Based Action Recognition: A Review of Recent Advances». In: *Neurocomput.* 537.C (mag. 2023), pp. 164–186. ISSN: 0925-2312 (cit. a p. 14).

[30] Chiara Plizzari, Marco Cannici e Matteo Matteucci. *Spatial Temporal Transformer Network for Skeleton-Based Action Recognition*. 2021 (cit. a p. 14).

[31] Amir Shahroudy et al. *NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis*. 2016. arXiv: `1604.02808 [cs.CV]` (cit. a p. 14).

[32] Jun Liu et al. «NTU RGBD 120: A Large-Scale Benchmark for 3D Human Activity Understanding». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.10 (ott. 2020), pp. 2684–2701 (cit. a p. 14).

[33] Avinandan Banerjee, Pawan Kumar Singh e Ram Sarkar. «Fuzzy Integral-Based CNN Classifier Fusion for 3D Skeleton Action Recognition». In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.6 (2021), pp. 2206–2216 (cit. a p. 14).

[34] Hossein Rahmani et al. *UWA 3D Multiview Activity II Dataset*. 2023 (cit. alle pp. 14, 15).

[35] Chen Chen, Roozbeh Jafari e Nasser Kehtarnavaz. «UTD-MHAD: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor». In: *2015 IEEE International Conference on Image Processing (ICIP)*. 2015, pp. 168–172 (cit. alle pp. 14, 15).

[36] Yinghao Huang et al. *Deep Inertial Poser: Learning to Reconstruct Human Pose from Sparse Inertial Measurements in Real Time*. 2018. arXiv: 1810.04703 [cs.GR] (cit. alle pp. 14, 15).

[37] Manuel Palermo et al. «From raw measurements to human pose - a dataset with low-cost and high-end inertial-magnetic sensor data». In: *Scientific Data* 9.1 (set. 2022), p. 591. ISSN: 2052-4463 (cit. alle pp. 14, 15).

[40] Jinhyeok Jang et al. *ETRI-Activity3D: A Large-Scale RGB-D Dataset for Robots to Recognize Daily Activities of the Elderly*. 2020. arXiv: 2003.01920 [cs.RO] (cit. a p. 15).

[41] Jiang wang et al. *Cross-view Action Modeling, Learning and Recognition*. 2014. arXiv: 1405.2941 [cs.CV] (cit. a p. 15).

[42] Daniela Micucci, Marco Mobilio e Paolo Napoletano. «UniMiB SHAR: A Dataset for Human Activity Recognition Using Acceleration Data from Smartphones». In: *Applied Sciences* 7.2017 (). ISSN: 2076-3417 (cit. a p. 15).

[43] Valentina Camomilla et al. «Trends Supporting the In-Field Use of Wearable Inertial Sensors for Sport Performance Evaluation: A Systematic Review». In: *Sensors* 18.3 (2018). ISSN: 1424-8220 (cit. a p. 15).

[44] Yue Luo et al. «A database of human gait performance on irregular and uneven surfaces collected by wearable sensors». In: *Scientific Data* 7.1 (lug. 2020), p. 219. ISSN: 2052-4463 (cit. a p. 15).

[45] Yuan-Pin Lin et al. *Multilayer perceptron for EEG signal classification during listening to emotional music*. 2007 (cit. a p. 16).

[46] Ojan Majidzadeh Gorjani et al. «Human Activity Classification Using Multilayer Perceptron». In: *Sensors* 21.18 (2021). ISSN: 1424-8220 (cit. a p. 16).

Capitolo 3

[47] Seyed Mostafa Hejazi e Charith Abhayaratne. «Handcrafted localized phase features for human action recognition». In: *Image and Vision Computing* 123 (2022), p. 104465. ISSN: 0262-8856 (cit. a p. 17).

[48] T. Ahmad e Moh Nasrul Aziz. «Data preprocessing and feature selection for machine learning intrusion detection systems». In: *ICIC Express Letters* 13 (gen. 2019), pp. 93–101 (cit. a p. 17).

[49] J.K. Aggarwal e M.S. Ryoo. «Human Activity Analysis: A Review». In: *ACM Comput. Surv.* 43.3 (apr. 2011). ISSN: 0360-0300 (cit. a p. 18).

[50] Tahera Hossain e Sozo Inoue. «A Comparative Study on Missing Data Handling Using Machine Learning for Human Activity Recognition». In: *2019 Joint 8th International Conference on Informatics, Electronics & Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*. 2019, pp. 124–129 (cit. a p. 20).

[51] Sotiris Kotsiantis, Dimitris Kanellopoulos e P. Pintelas. «Data Preprocessing for Supervised Learning». In: *International Journal of Computer Science* 1 (gen. 2006), pp. 111–117 (cit. a p. 20).

[53] Abdul Kadar Muhammad Masum et al. «A Statistical and Deep Learning Approach for Human Activity Recognition». In: *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*. 2019, pp. 1332–1337 (cit. a p. 23).

Capitolo 5

[58] Angelos Karatsidis et al. «Estimation of Ground Reaction Forces and Moments During Gait Using Only Inertial Motion Capture». In: *Sensors* 17.1 (2017). ISSN: 1424-8220 (cit. a p. 32).

[65] W.T. Cochran et al. «What is the fast Fourier transform?» In: *Proceedings of the IEEE* 55.10 (1967), pp. 1664–1674 (cit. a p. 42).

Sitografia

Capitolo 1

[5] *DrapeBot*. URL: https://www.drapebot.eu/ (cit. alle pp. 3, 52).

[7] *Franka Emika*. URL: https://www.franka.de/ (cit. a p. 4).

[8] *Universal Robots*. URL: https://www.universal-robots.com/ (cit. a p. 4).

Capitolo 2

[15] *IBM - Recurrent neural networks*. URL: https://www.ibm.com/it-it/topics/recurrent-neural-networks (cit. a p. 13).

[26] *Transformers explained*. URL: https://towardsdatascience.com/transformers-141e32e69591 (cit. a p. 13).

[38] *NTU dataset by ROSE Lab*. URL: https://rose1.ntu.edu.sg/dataset/actionRecognition/ (cit. a p. 14).

[39] *UTD Multimodal Human Action Dataset (UTD-MHAD)*. URL: https://personal.utdallas.edu/~kehtar/UTD-MHAD.html (cit. a p. 15).

[57] *Xsens positioning explained*. URL: https://base.xsens.com/s/article/Sensor-Placement-in-Xsens-Awinda-System (cit. alle pp. 31, 37).

Capitolo 3

[52] *Levity.ai*. URL: https://levity.ai/ (cit. a p. 22).

[54] *3Blue1Brown*. URL: https://www.3blue1brown.com/ (cit. a p. 24).

Capitolo 5

[55] *NASA, "Towards Autonomous Operation of Robonaut 2"*. URL: https://ntrs.nasa.gov/citations/20110024047 (cit. a p. 28).

[56] *OpenPTrack*. URL: http://openptrack.org/ (cit. a p. 29).

[59] *Xsens*. URL: https://www.movella.com/products/xsens (cit. a p. 31).

[61] *Edge impulse*. URL: https://www.edgeimpulse.com/ (cit. alle pp. 33, 41, 42).

[62] *IAS-Lab - Università di Padova*. URL: http://robotics.dei.unipd.it/ (cit. a p. 35).

[63] *Videocamera Intel realsense d455*. URL: https://www.intelrealsense.com/depth-camera-d455/ (cit. a p. 38).

[64] *OpenPTrack v2 GitHub*. URL: https://github.com/OpenPTrack/open_ptrack_v2 (cit. a p. 39).

[66] *Why ROS's timestamps are not enough*. URL: https://pkok.github.io/2014/10/16/#single-sensor-setting (cit. a p. 49).

Appendice Codice

Programma C++ che converte un file Rosbag contenente i dati dei giunti in file csv

```
#include <ros/ros.h>
#include <rosbag/view.h>
#include <dirent.h>
#include <sys/stat.h>

#include <iostream>
#include <fstream>

#include <gtest/gtest.h>
#include <opt_msgs/SkeletonTrackArray.h>
#include <boost/foreach.hpp>

#include <vector>
#include <string>
#include <algorithm>

#define foreach BOOST_FOREACH

int main() {
    rosbag::Bag bag;
    std::string path;
    std::string dir_path;
    std::string file_name;
    std::vector<std::string> files;
    std::vector<int> skeleton_used = {169, 190, 205, 220, 282, 292, 299, 326,
    336, 365, 378, 848, 912, 205, 912, 960, 999, 1055, 1376, 1528, 1627, 1797,
     1801 };

    Track3D track3D;

```

```
29      std::cout << "Inserire il percorso della cartella dei file da convertire,
        partire dalla cartella /home" << std::endl;
30      std::cin >> dir_path;
31
32      DIR *dir; //pointer to directory
33      struct dirent *ent; //pointer to directory entry
34      struct stat st; //structure for file status
35
36      if((dir = opendir(dir_path.c_str())) != NULL) {
37          while ((ent = readdir(dir)) != NULL) {
38              std::string file_path = dir_path + ent->d_name; //create full file
         path by appending directory path and file name
39              stat(file_path.c_str(), &st); //get file status and save it in st
40
41              if (S_ISREG(st.st_mode)) {                    //check if the entry
        is a regular file
42                  files.push_back(ent->d_name);           //add file name to
        vector
43              }
44          }
45          closedir(dir); //close the directory
46      }
47      else {
48          std::cout << "Non e' stato possibile aprire la directory" << std::endl
        ;
49          return 1;
50      }
51
52      //creation of files csv using the list of files produced before
53      while(!files.empty()) {
54          std::string line;
55          std::string clean;
56          std::string temp;
57          std::string first_line = "timestamp,";
58
59          int timestamp = 0;
60          double start = 0;
61          file_name = files.at(files.size()-1);
62          files.pop_back();
63          path = dir_path + file_name;
64          file_name = file_name.substr(0,13) + "csv";
65          std::cout << file_name << std::endl;
66          bag.open(path, rosbag::bagmode::Read);
67          rosbag::View view(bag);
```

```
68
69          std::ofstream file(file_name);
70
71          if(!file.is_open()) {
72              std::cerr << "Error, was not able to open the file!" << std::endl;
73              return 1;
74          }
75
76          //define the first line of the .csv file containing the fields name
77          for(int i = 0; i < 15; i++ ) {
78              first_line = first_line + std::to_string(i) + "_position_x," +
79              std::to_string(i) + "_position_y," +
80              std::to_string(i) + "_position_z";
81              if(i != 14)
82                  first_line = first_line + ",";
83          }
84          //write the first line into the file
85          file << first_line << std::endl;
86
87          //scan every timestamp
88          foreach(rosbag::MessageInstance const m, view) {
89              opt_msgs::SkeletonTrackArray::ConstPtr s = m.instantiate<opt_msgs
    ::SkeletonTrackArray>();
90
91              if (s != NULL){
92                  temp = std::to_string(timestamp*33); //milliseconds
93                  int j = 0;
94
95                  //skip the first message
96                  if (s->tracks.size() == 0)
97                      continue;
98
99                  //looks for the track in the array of accepted values
100                 auto it = std::find(skeleton_used.begin(), skeleton_used.end()
    , s->tracks[j].id);
101
102                 while(it == skeleton_used.end()){
103                     if(j < s->tracks.size()-1)
104                         j++;
105                     else
106                         j = 0;
107                     it = std::find(skeleton_used.begin(), skeleton_used.end(),
     s->tracks[j].id);
108                 }
```

```
109
110                 //produces the line corresponding to the current sample
111                 for(int i = 0; i < 15; i++)
112                     temp = temp + "," + std::to_string(s->tracks[j].joints[i].
    x) + "," + std::to_string(s->tracks[j].joints[i].y) + "," + std::to_string
    (s->tracks[j].joints[i].z);
113
114                 //insert the line produced into the file
115                 file << temp << std::endl;
116                 temp = "";
117                 timestamp++;
118             }
119         }
120
121         file.close();
122         bag.close();
123     }
124     return 0;
125 }
```

Codice 6.1: rgbd_skeleton.cpp