UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI TECNICA E GESTIONE DEI SISTEMI INDUSTRIALI
CORSO DI LAUREA MAGISTRALE IN INGEGNERIA MECCATRONICA

TESI DI LAUREA MAGISTRALE

Anticipazione del movimento umano attraverso previsione multi-futuro di traiettorie 3D vincolate

Relatore: Prof. Stefano Michieletto

Laureando: Matteo Cunico
matr. 1241867

ANNO ACCADEMICO: 2021-22

SOMMARIO

La modellazione e l'anticipazione del movimento umano costituiscono due attività fondamentali in diversi ambiti di ricerca, tra i quali la robotica collaborativa, la guida autonoma e i sistemi di sorveglianza intelligente. Per la risoluzione di questi task, si sono imposte negli ultimi anni diverse metodologie basate sulle reti neurali, anche grazie allo sviluppo relativamente recente di tecniche sempre più raffinate di machine learning (*apprendimento automatico*), e in particolare di deep learning (*apprendimento profondo*).
Il lavoro presentato in questa tesi si inserisce in tale contesto. Viene in particolare proposto un metodo che permette di generare una o più predizioni relative ai movimenti futuri di una persona a partire da quelli passati. Per fare questo, il sistema inizialmente scompone la catena cinematica del corpo umano in ciascuno dei suoi giunti (ginocchio, spalla, polso, ...), trattando il movimento del soggetto come un insieme di traiettorie percorse da ognuno di questi punti. Per ogni traiettoria di giunto osservata, uno specifico modello di predizione (basato sulle reti neurali) si occupa di fornire una pluralità di traiettorie future plausibili. Queste ultime vengono in seguito combinate, allo scopo di ritornare ad una rappresentazione globale del corpo, ed infine vincolate per evitare di ottenere previsioni con pose umane future non possibili.
Per la sperimentazione dell'algoritmo si è utilizzato il dataset AMASS, che include al suo interno una notevole quantità di dati relativi ai movimenti di soggetti umani in scene tridimensionali. I risultati che si ottengono sono in linea con i lavori già esistenti, fatto che permette di affermare che il sistema di predizione proposto risulta efficace nello svolgere il compito richiesto.

INDICE

ELENCO DELLE FIGURE

ELENCO DELLE TABELLE

INTRODUZIONE

Nella vita di tutti i giorni sfruttiamo in maniera più o meno consapevole la nostra capacità di anticipare i movimenti futuri delle persone con cui interagiamo. Quando camminiamo in una piazza affollata, ad esempio, siamo in grado di riconoscere in anticipo quale percorso intraprendere in modo da evitare di collidere con altri individui. Oppure, nel caso in cui dobbiamo consegnare un oggetto nelle mani di un'altra persona, abbiamo un'idea abbastanza precisa di quando possiamo rilasciare la presa. Quest'abilità di predizione è alla base della vita sociale di esseri umani e animali. Allo stesso modo, tutte le macchine o i sistemi automatici concepiti per interagire o collaborare con le persone necessitano di una simile predisposizione.
L'anticipazione del movimento umano è un compito complesso ma di primaria importanza per diversi ambiti applicativi, tra i quali:

- la guida autonoma [27], in cui è necessario implementare sistemi in grado di stimare in anticipo la posizione futura dei pedoni circostanti;
- la sorveglianza intelligente [18], nella quale si studia la possibilità di prevedere incidenti o situazioni critiche (ad esempio un furto con scasso o incidenti di caduta di persone anziane) a partire dalle immagini fornite da una telecamera;
- il tracciamento di soggetti umani in ambienti caratterizzati da occlusioni [35, 28];
- il settore dell'assistenza sanitaria [40], che può trarne vantaggio a fini diagnostici;
- la collaborazione uomo-robot (HRC, *Human-Robot Collaboration*) [17, 39, 36], campo di ricerca che mira a realizzare degli algoritmi che forniscano ai robot la capacità di collaborare con l'uomo (si parla infatti di robot collaborativi o *cobot*), permettendogli così di svolgere delle attività in cooperazione con un certo numero di esseri umani, dal contesto industriale a quello domestico, in uno spazio di lavoro condiviso.

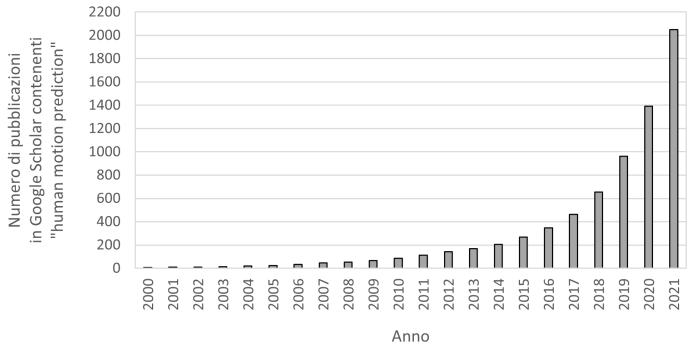

Figura 1: Andamento annuale del numero di pubblicazioni relative all'argomento "human motion prediction" (dati Google Scholar).

In particolare, l'ultimo punto è quello che probabilmente ha spinto di più nello sviluppo di algoritmi di previsione del movimento umano. Nella stragrande maggioranza degli attuali scenari industriali, i cobot lavorano nello stesso spazio di lavoro destinato agli operatori umani, ma in modalità sequenziale. Se viene rilevato un eccessivo aumento del rischio di collisione, il robot si blocca in attesa che la persona sia uscita dalla zona di pericolo. La completa collaborazione (e non una semplice interazione) tra un essere umano ed una macchina per l'esecuzione di un compito non può avvenire in maniera sicura senza che il robot sia in grado di anticipare i movimenti dell'operatore umano, adattando le proprie azioni future di conseguenza.
L'interesse per questo argomento ha conosciuto una notevole crescita negli ultimi anni, documentata anche dall'aumento esponenziale del numero di pubblicazioni correlate (Figura 1).
L'anticipazione del moto è un problema estremamente complesso. Innanzitutto, la dinamica biomeccanica del movimento umano dipende da un numero elevato di parametri ed ha un carattere fortemente non lineare, fattori che ne rendono assai difficile una modellazione realmente accurata. Inoltre, per ampi orizzonti di tempo, il fenomeno del movimento umano assume una natura stocastica con un alto livello di incertezza.
Per i motivi sopra citati, i tradizionali approcci alla soluzione del problema dell'anticipazione sfruttano modelli di tipo probabilistico, basati su processi stocastici bayesiani e/o gaussiani [42, 19] o sugli hidden Markov models [3]. Recentemente, grazie all'avanzamento delle tecnologie di Deep Learning, si sono imposte le reti neurali, ad oggi i sistemi più diffusi nello svolgimento di questo compito

[2, 25, 8, 16, 5, 9]. Le reti neurali, infatti, consentono di risolvere il problema con un approccio di tipo "black box", senza cioè dover conoscere a priori la struttura del modello reale del corpo umano e dei suoi movimenti.
Il lavoro presentato in questa tesi si inserisce in tale contesto. Lo scopo è l'implementazione di un sistema di anticipazione che, noti i movimenti precedenti di una persona, sia in grado di generare un certo numero di scenari di movimento futuri per il soggetto in esame. Visto l'elevato grado di incertezza, occorre che la predizione non sia a singolo futuro, ma *multi-futuro*: i futuri generati devono essere molteplici e differenti tra loro. Nella strategia di predizione proposta nel seguito, il corpo umano viene trattato come un meccanismo articolato costituito da un certo numero di giunti ed organi terminali. Ad ogni istante viene identificata l'intera posa (o postura) del corpo, intesa come la configurazione di tutti i suoi giunti, oltre che la sua posizione assoluta. Una movimentazione tridimensionale può essere quindi trattata come un insieme di traiettorie 3D, ognuna relativa ad un giunto del corpo. In altre parole, il problema dell'anticipazione del movimento di una persona viene scisso in una serie di sotto-problemi di predizione di traiettorie, ovvero di punti che si spostano in un particolare spazio. Nello specifico, le coordinate 3D della posizione assoluta del corpo sono riferite allo spazio metrico, mentre i punti 3D che esprimono le orientazioni dei giunti sono definiti nello spazio della notazione asse-angolo (che è la modalità di rappresentazione delle rotazioni scelta in questo lavoro). L'algoritmo di previsione multi-futuro delle singole traiettorie trae ispirazione da Multiverse [20], un modello basato sulle reti neurali inizialmente progettato per lavorare con generiche scene video 2D derivanti da telecamere per la videosorveglianza pubblica. Ogni giunto viene elaborato in maniera indipendente dagli altri, approccio che rende necessario un controllo ed un eventuale aggiustamento a posteriori qualora si riscontrassero predizioni di movimenti non realmente riproducibili da un essere umano. A questo scopo, a tutte le traiettorie predette sono applicati dei ben precisi vincoli, riferiti ai limiti intrinseci del corpo umano. Un supporto nella definizione di questi ultimi è offerto dal software OpenSim [6], impiegato in biomeccanica per l'analisi e la simulazione del movimento.
Il documento è strutturato come segue. Il Capitolo 1 definisce ed espone in maniera dettagliata il problema dell'anticipazione, con riferimenti ad altri lavori esistenti. Nel Capitolo 2 viene fornita una

panoramica sulle reti neurali, focalizzata in particolare sulle architetture utilizzate in questo lavoro. Il Capitolo 3 presenta minuziosamente il modello di anticipazione utilizzato in questo lavoro ed il principio di funzionamento di ogni sua componente. Nel Capitolo 4, infine, sono riportati i risultati ottenuti testando il sistema con un dataset pubblico (AMASS [23]), con una discussione degli stessi e dei possibili miglioramenti futuri.

1

ANTICIPAZIONE DEL MOVIMENTO UMANO

1.1 DEFINIZIONE DEL PROBLEMA

I movimenti che una persona compie all'interno di una scena tridimensionale possono essere scomposti in una sequenza temporale discreta di *pose* collocate nello spazio. Una posa, da intendersi come sinonimo di postura, è descrivibile con un certo numero di parametri (N, esposti nella Sezione 1.3), dunque con un vettore $\boldsymbol{P} \in \mathbb{R}^N$. Di conseguenza, un movimento umano è rappresentato dalla successione $\boldsymbol{P}_{1:T} = \{\boldsymbol{P}_1, \boldsymbol{P}_2, ..., \boldsymbol{P}_T\}$, dove $1, 2, ..., T$ sono gli istanti di tempo di cui si tiene conto.
Il problema dell'anticipazione del movimento umano si pone nel modo seguente: dato il campione di movimento $\boldsymbol{P}_{1:T_o}$, nel quale sono memorizzate le ultime T_o pose osservate, ricavare un certo numero (F_B) di movimenti futuri possibili contenenti T_p pose. L'output è quindi una sequenza $\hat{\boldsymbol{P}}^{1:F_B}_{T_o+1:T_o+T_p} = \{\hat{\boldsymbol{P}}^{1:F_B}_{T_o+1}, ..., \hat{\boldsymbol{P}}^{1:F_B}_{T_o+T_p}\}$, dove $\hat{\boldsymbol{P}}^{1:F_B}_t = \{\hat{\boldsymbol{P}}^1_t, ..., \hat{\boldsymbol{P}}^{F_B}_t\}$ contiene tutte le F_B possibili pose predette all'istante t. T_o denota l'orizzonte di osservazione, mentre T_p è l'orizzonte di predizione.
Le difficoltà di quest'operazione sono molteplici. Innanzitutto, il movimento umano è molto complesso e dipende da un gran numero di parametri. Pertanto, N deve essere abbastanza alto da garantire una corretta modellazione del corpo umano, anche in funzione del tipo di rappresentazione utilizzata per quest'ultimo, rendendo $\mathbb{R}^N$ (lo spazio delle pose) uno spazio ad elevata dimensionalità. Inoltre, la non linearità del fenomeno complica ulteriormente il compito di modellazione. A queste problematiche va aggiunta anche la natura stocastica del movimento umano, dominata da un alto grado di incertezza. I movimenti futuri di una persona dipendono infatti da tre fattori: i movimenti precedenti, l'ambiente circostante (presenza di ostacoli o vie preferenziali) e le intenzioni del soggetto (ad esempio raccogliere un oggetto da terra oppure uscire da una stanza). A differenza dei primi due punti, l'ultimo è molto difficile da modellare in maniera algoritmica. In questo lavoro ci si focalizzerà sul primo, ovvero sull'implementazione di un sistema di predizione che riceva in ingresso i

dati sui movimenti precedenti del soggetto umano in modo da fornire in uscita F_B movimenti futuri possibili, trascurando le informazioni relative al contesto limitrofo o alle intenzioni dell'individuo.
Incrementando l'orizzonte di predizione, aumenta l'incertezza del moto, dunque il task dell'anticipazione si fa sempre più impegnativo. In base alla durata (in secondi) dei movimenti predetti, si distinguono due tipi di predizione: a breve e a lungo termine. Non esiste una netta separazione tra le due categorie, anche se spesso un orizzonte più lungo di 1 secondo viene considerato come lungo termine.

1.2 LAVORI CORRELATI

Come già spiegato nell'introduzione, la quasi totalità dei sistemi attuali per l'anticipazione del movimento umano impiega modelli basati su reti neurali.
Nel 2015, Fragkiadaki et al. [8] proposero il modello di predizione ERD (Encoder-Recurrent-Decoder) fondato su uno schema encoder-decoder, con uno strato intermedio costituito da reti neurali ricorrenti. Il modello ERD riceve in ingresso un vettore indicante la posa di un soggetto umano al passo temporale t, restituendo in uscita una predizione riguardante la posa al passo $t+1$. L'encoder si occupa di "mappare" la posa di input in uno spazio latente a dimensione ridotta, mentre il decoder effettua la trasformazione inversa. La RNN centrale viene allenata in modo da approssimare una funzione che correli la rappresentazione latente al passo t a quella al passo $t+1$. Una volta ricavata questa funzione, dalla posa al passo $t+1$ si può ottenere quella al passo $t+2$ e così via, generando intere sequenze future di movimento. Questo approccio, tuttavia, risente eccessivamente dell'accumulazione dell'errore nel tempo, rendendolo adatto esclusivamente per predizioni di breve termine (problema studiato da Ghosh et al. nel 2017 [9]). Un altra criticità di questo metodo è che vengono implicitamente considerate unicamente le dipendenze temporali dei movimenti, mentre sono ignorate le correlazioni spaziali tra le diverse parti del corpo.
Sotto quest'ultimo punto di vista, una soluzione migliore venne fornita sempre nel 2015 da Jain et al. [16], i quali suggerirono di sfruttare dei sistemi di predizione (sempre basati sulle RNN) strutturati in modo simile al corpo umano, definendo così il concetto di previsione *strutturata* del movimento. Il principale incoveniente è dato dal fatto

che questa struttura è piuttosto complessa e deve essere definita manualmente.
Sempre nell'ottica di una predizione strutturata, nel 2017 Bütepage et al. [5] proposero una diversa tipologia di rete encoder-decoder, con la struttura gerarchica del corpo umano direttamente incorporata nell'encoder.
Nello stesso anno, Martinez et al. [25] presentarono un modello basato sull'architettura sequence-to-sequence, la cui novità fondamentale era che la previsione non avveniva sulla posizione dei giunti del corpo, ma sulla loro velocità. Lo scopo era quello di ottenere delle traiettorie "filtrate", ovvero più lisce e regolari, rispetto ai lavori precedenti. Nella loro pubblicazione venne riportato anche un altro fatto interessante. Definirono un modello denominandolo *Zero-Velocity*, nel quale l'ultima posa osservata veniva replicata per un certo numero di volte e fornita in output come sequenza di movimento predetta. Questo approccio, ovviamente inadeguato alla risoluzione del problema, forniva comunque risultati quantitativi migliori rispetto ai lavori al tempo esistenti. Da quel momento, il modello Zero-Velocity è considerato come un punto di riferimento per la valutazione della qualità delle previsioni.

1.3 MODELLAZIONE DEL MOVIMENTO

Per modellare i movimenti di una persona all'interno di una scena tridimensionale, occorre dapprima introdurre una modalità di rappresentazione dei dati riguardanti le possibili pose del soggetto.
Nel corso degli anni sono stati proposti differenti approcci, ma il più comune utilizza una rappresentazione cinematica dello scheletro, modellato come un meccanismo articolato dotato di un certo numero di membri rigidi di lunghezza nota (che riproducono le ossa) vincolati tra loro da giunti di tipo rotoidale (le articolazioni) [13]. Il corpo umano, in realtà, è costituito da centinaia di giunti. È necessario selezionarne solo un sottoinsieme, che sia comunque sufficiente a rappresentare una posa in maniera adeguatamente accurata. In questo lavoro si utilizza lo scheletro del modello *SMPL* (*Skinned Multi-Person Linear model*) [21], illustrato in Figura 2. Esso è composto da 23 membri rigidi collegati mediante 18 giunti (segnati in nero in Figura 2). Sono distinguibili 5 catene cinematiche dotate di altrettanti organi terminali (end-effectors):

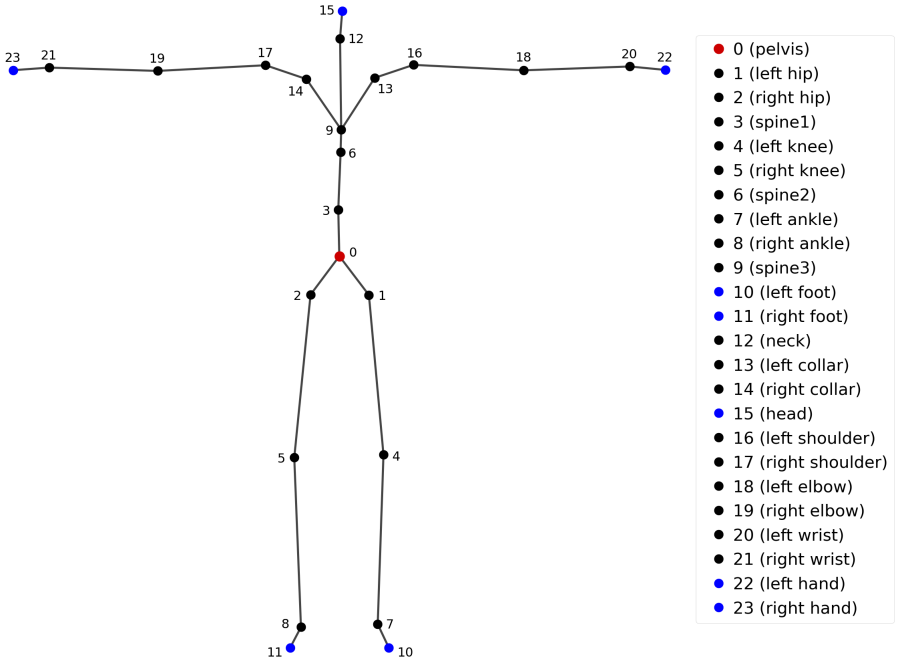

Figura 2: Modello scheletrico del corpo umano (SMPL).

mani, piedi e testa (in blu).

Con la rappresentazione scheletrica del corpo umano, una posa è identificabile esplicitando la posizione nello spazio di tutti i giunti e degli organi terminali (come in [5]). Nell'ambito dell'anticipazione del movimento, però, questo approccio può essere problematico. Una rappresentazione "posizionale", infatti, ha il difetto di essere ridondante sui vincoli di lunghezza dei membri. Un algoritmo che si occupa di prevedere le posizioni 3D di tutti i giunti dovrebbe includere un complesso sistema di vincoli non lineari per non incorrere in fenomeni di contrazione o dilatazione dei membri (rigidi per ipotesi). Un altro svantaggio è legato al fatto che la conoscenza delle posizioni non include alcuna informazione riguardo all'orientazione dei giunti e degli organi terminali.

Per risolvere la questione, è possibile avvalersi di un altro tipo di rappresentazione. In [16] viene definito un sistema di riferimento XYZ per ciascuno dei giunti, permettendo di identificare una posa indicando per ogni giunto la rotazione 3D relativa rispetto al sistema di riferimento del "giunto padre". Un ulteriore sistema XYZ viene specificato in corrispondenza di un particolare keypoint in corrispondenza del bacino, denominato *root* poiché costituisce la "radice" della catena cinematica dello scheletro umano (indicato in rosso in Figura 2). Quest'ultimo sistema di riferimento consente di collocare nello spazio

l'intero corpo umano, in termini di: (*i*) orientazione, attraverso una rotazione rispetto ad un sistema di riferimento fisso (assoluto), e (*ii*) posizione, indicando le coordinate del root rispetto al sistema di riferimento assoluto. Questo criterio ha il vantaggio di essere invariante rispetto sia alla locazione del soggetto umano, sia alle sue dimensioni fisiche, poichè le lunghezze dei membri sono ininfluenti. Conoscendo queste ultime, si può agevolmente passare alla rappresentazione posizionale tramite l'analisi cinematica diretta. Il procedimento inverso, l'analisi cinematica inversa, è invece alquanto laborioso e necessita di essere risolto con metodi numerici. Per i motivi sopra descritti, in questo lavoro si è preferito rappresentare le pose indicando le rotazioni dei giunti, anzichè le posizioni.
Lo scheletro SMPL, come già scritto, è dotato di $J = 18$ giunti e $N_{ee} = 5$ organi terminali, a cui si aggiunge il root keypoint. Il numero di keypoint nel corpo umano dei quali siamo interessati a conoscere la posizione è $N_{kp} = J + N_{ee} + 1 = 24$. Il numero di parametri necessari ad identificare una posa 3D, invece, è pari a $N = N_{pr} \cdot (J + 1) + 3$, essendo N_{pr} e 3 il numero di parametri rispettivamente per le rotazioni e per le traslazioni.
N_{pr} (maggiore o uguale a 3) dipende dal tipo di notazione scelta per le rotazioni: $N_{pr} = 3$ utilizzando gli angoli di Eulero, $N_{pr} = 4$ con i quaternioni, $N_{pr} = 9$ con le matrici di rotazione. Utilizzando una rappresentazione delle rotazioni con $N_{pr} > 3$, si rende necessaria l'applicazione di particolari vincoli, tipicamente non lineari, per portare a 3 l'effettivo numero di gradi di libertà della rotazione 3D. Una rappresentazione minima (con $N_{pr} = 3$) permette di evitare questa complicazione. D'altra parte, gli angoli di Eulero sono spesso causa di errori, perchè: (*i*) occorre sempre specificare la sequenza delle tre rotazioni elementari, (*ii*) una rotazione non si può esprimere univocamente con una terna e (*iii*) soffrono del cosiddetto *blocco cardanico* (più noto con il termine inglese *gimbal lock*), inteso come la perdita di un grado di libertà rotazionale a seguito dell'allineamento di due assi rotanti verso la stessa direzione. Questi aspetti sono molto critici soprattutto per modelli di predizione basati su reti neurali. Un'alternativa, che verrà sfruttata nel seguito, è la cosiddetta rappresentazione *asse-angolo*, nota anche come *vettore di rotazione* o *mappa esponenziale* [11]. Con questa notazione, una rotazione 3D di misura θ attorno ad un asse parallelo al versore $\boldsymbol{k}$ viene rappresentata con il vettore $\boldsymbol{v} = \theta \cdot \boldsymbol{k}$ ($\boldsymbol{v} \in \mathbb{R}^3$). In questo caso, l'unica ambiguità si ha in corrispondenza

delle rotazioni di misura $\theta = \pm\pi$, poiché la rotazione individuata dalla coppia versore-misura $(\boldsymbol{k}, \pi)$ è identica a quella data da $(\boldsymbol{k}, -\pi)$. In conclusione, una posa viene descritta con $N = 3 \cdot (J + 2) = 60$ parametri divisi in $J + 2 = 20$ terne ($J + 1 = 19$ rotazioni e 1 traslazione). Le dimensioni fisiche dello scheletro "base" di SMPL sono relative ad un essere umano di 170 cm di stazza media ($\sim$ 65 kg). A partire da questo si possono impiegare 16 coefficienti "di forma", i quali in poche parole esprimono delle dilatazioni o delle compressioni rispetto alle misure iniziali, per definire in maniera più precisa l'effettiva corporatura di un certo individuo. Ovviamente, questi coefficienti di forma sono caratteristici di ciascuna persona, pertanto non verranno presi in considerazione in questa tesi.
Il movimento umano è soggetto a dei vincoli di natura anatomica, quindi non tutte le configurazioni di giunti sono associate a pose ammissibili. Ad esempio, le ginocchia possono ruotare con facilità attorno ad un asse (flessione), mentre le rotazioni attorno ad altri assi sono trascurabili, e di conseguenza possiamo considerarle come inibite. L'effettivo numero di gradi di libertà (GdL) di una posa è quindi minore di N. Altri giunti sono più liberi, come le anche che consentono tre rotazioni elementari: flessione, adduzione e torsione. In Tabella 1 sono riportati i gradi di libertà effettivi per ogni giunto dello scheletro SMPL. Il totale è di 48, inclusi i 6 della posizione e dell'orientazione assolute del corpo. Ogni rotazione permessa da qualunque

Indici	Giunto	Num. GdL
1,2	anca	3
3,6,9	colonna vertebrale	3
4,5	ginocchio	1
7,8	caviglia	2
12	collo	3
13,14	clavicola	2
16,17	spalla	3
18,19	gomito	2
20,21	polso	2
	Totale	**42**

Tabella 1: Gradi di libertà effettivi per ogni giunto dello scheletro SMPL.

giunto, comunque, ha dei limiti. Non è possibile, per esempio, flettere in avanti il ginocchio o torcere l'anca oltre una certa misura. Se le rotazioni sono espresse con gli angoli di Eulero, questo tipo di vincolo si traduce come $\Theta_{lb} \leq \theta \leq \Theta_{ub}$ (dove θ è una generica coordinata libera di un giunto, la quale varia entro l'intervallo compreso tra Θ_{lb} e Θ_{ub}). I limiti inferiore e superiore Θ_{lb} e Θ_{ub} non sono semplici da ricavare, perché sono tutt'altro che univoci. Essi infatti variano con l'età, il sesso, la condizione fisica e molti altri fattori.

1.4 IL DATASET AMASS

Per allenare (e in seguito testare) un modello di anticipazione dei movimenti umani basato sulle reti neurali, occorre disporre di una gran mole di dati relativi a pose di persone. Esistono molti dataset contenenti pose e movimenti umani 3D, ma il più vasto è *AMASS* (Archive of Motion capture As Surface Shapes) [23]. AMASS, rilasciato nel 2019, è il risultato della composizione e della rielaborazione di dataset già esistenti e disponibili, tra cui *BMLrub* [40], *KIT* [24] e *MPI HDM05* [26] (per citare quelli più corposi). Complessivamente, sono forniti 11265 campioni di movimento 3D (sotto forma di altrettanti "motion files") relativi a 344 soggetti (umani) distinti. Il tempo totale di registrazione è di 2420 minuti (una media di 12.9 secondi per motion file). I dati sono campionati a 60 Hz, il che porta ad avere circa 8.7 milioni di frame (pose 3D) disponibili (774 per motion file).
Ogni motion file di AMASS contiene le informazioni riguardanti un campione di movimento di una persona (cammino, corsa, salto, ...) all'interno di una area circoscritta. I dati delle pose 3D sono espressi nei formati *SMPL+H* [33] (estensione di SMPL che include la modellazione dei giunti delle mani, per un totale di $J = 51$ giunti) e *SMPL-X* [29] (altra estensione di SMPL rivolta alle espressioni facciali, con $J = 53$ giunti complessivi). Una movimentazione di durata T frame è memorizzata in due array (entrambi serializzati nel motion file): uno di dimensione $T \times 3$ che denota l'andamento delle coordinate spaziali del root keypoint, l'altro di dimensione $T \times (3\,(J+1))$ che indica, in notazione asse-angolo, i parametri delle rotazioni 3D dei giunti e dell'orientazione dell'intero corpo. Per gli scopi di questo lavoro, solo le prime 22 rotazioni (66 parametri) di SMPL+H o SMPL-X sono di interesse, perché relative ai giunti dello scheletro SMPL base. Si pos-

sono inoltre scartare le rotazioni in corrispondenza degli indici degli organi terminali SMPL: le caviglie (10 e 11) e la testa (15, necessaria per SMPL-X).

2

RETI NEURALI

Le reti neurali sono complessi sistemi di calcolo che tentano di imitare la straordinaria capacità di elaborazione del cervello umano. Esse hanno la facoltà di "imparare" a risolvere un problema a seguito di un opportuno "allenamento", e costituiscono il fondamento delle attuali tecniche di apprendimento automatico.
Nel presente capitolo sono presentati i concetti fondamentali relativi a questi sistemi.

2.1 GENERALITÀ

Il cervello umano è costituito da un elevato numero (circa 10^{11}) di unità elementari densamente interconnesse tra di loro: i neuroni. Ognuna di queste entità è fisicamente collegata a molti (circa 10^4) altri neuroni con i quali è in grado di scambiare informazioni sotto forma di impulsi elettrici [12].

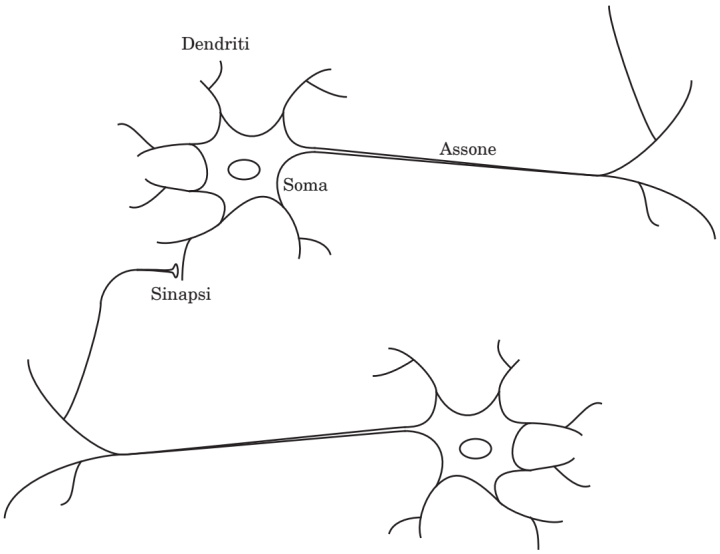

Figura 3: Schema semplificato di un neurone biologico [12].

La struttura semplificata di un neurone (Figura 3) si compone di tre parti fondamentali:

- il soma, ovvero il corpo cellulare, deputato alle principali funzioni della cellula;
- l'assone, incaricato della propagazione degli impulsi nervosi verso altri neuroni. Rappresenta l'output della cellula, è unico ma si dirama in molteplici direzioni;
- i dendriti, connessi per mezzo di giunzioni sinaptiche agli assoni di altri neuroni, dai quali ricevono gli impulsi. Veicolano i numerosi segnali in input alla cellula.

Un neurone biologico costituisce quindi il nodo elementare di una rete molto più complessa, nota come rete neurale biologica. La funzione di una rete neurale è determinata dalla disposizione dei singoli neuroni e dalla "forza" di ciascun legame. Alcune strutture sono definite già alla nascita, mentre altre si sviluppano e variano durante tutto il corso della vita. Di fatto, il processo di apprendimento avviene a seguito della continua modifica, intesa come rafforzamento o indebolimento, delle varie interconnessioni tra neuroni.
Le reti neurali artificiali (note semplicemente come reti neurali, in inglese *neural networks*, NN) sono dei modelli di calcolo strutturati in maniera approssimativamente simile alle reti neurali biologiche. Il loro concepimento rappresenta il tentativo di emulare, per mezzo di un calcolatore, il funzionamento di un cervello umano o animale. La peculiarità di questi sistemi è la capacità di apprendere, in analogia alla controparte biologica, come svolgere un dato compito sulla base dell'esperienza precedente, senza quindi la necessità di implementare regole specifiche per tale attività. I recenti sviluppi di ricerca hanno notevolmente ampliato i campi di applicazione delle NN, impiegate nella risoluzione di molti problemi di intelligenza artificiale legati all'informatica, all'elettronica, all'automotive, alla medicina e a numerosi altri settori.
Il nodo elementare di una rete neurale artificiale è, per l'appunto, il neurone artificiale (Figura 4), modellato come un sistema con m ingressi (x_1, x_2, ..., x_m) e una sola uscita (y_k). Ogni connessione tra ingressi e neurone è caratterizzata da un certo peso ($w_{k,1}$, ..., $w_{k,m}$). L'uscita è esprimibile in termini matematici come:

$$y_k = \phi\left(\sum_{i=1}^{m} w_{k,i} x_i + b_k\right) \tag{1}$$

dove b_k denota un bias (o offset) e $\phi(\cdot)$ è la cosidetta *funzione di attivazione* del neurone. Quest'ultima definisce come e per quale configura-

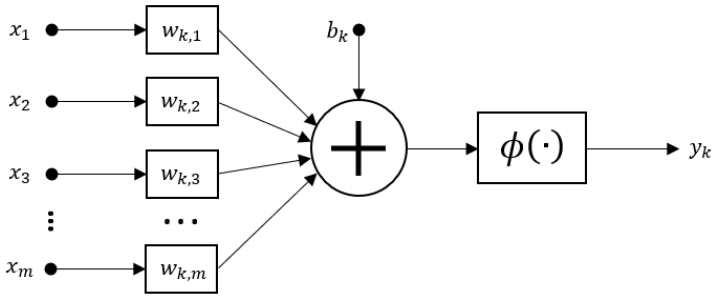

Figura 4: Modello di un neurone artificiale.

zione di ingressi il nodo deve propagare il segnale all'uscita, ovvero si deve "attivare". Tipicamente, le funzioni di attivazione restituiscono valori compresi tra 0 e 1 (OFF/ON) ed hanno un carattere fortemente non lineare. Esempi di funzioni di attivazione sono (Figura 5):

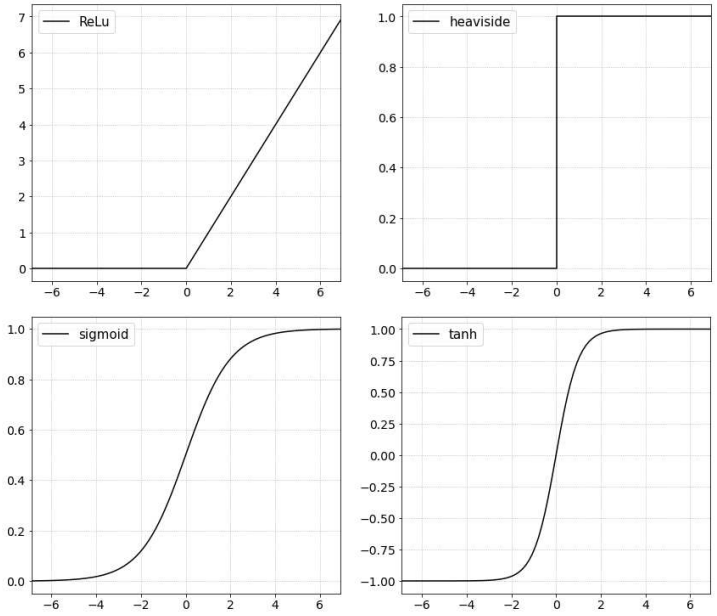

Figura 5: Alcune funzioni di attivazione.

- il rettificatore (ReLU, *rectified linear unit*):

$$\phi(u) = \begin{cases} u & \text{se } u > 0 \\ 0 & \text{se } u \leq 0 \end{cases} \tag{2}$$

- il gradino di Heaviside:

$$\phi(u) = \begin{cases} 1 & \text{se } u \geq 0 \\ 0 & \text{se } u < 0 \end{cases} \tag{3}$$

- la funzione sigmoidea:

$$\phi(u) = \frac{1}{1 + e^{-u}} \tag{4}$$

- la tangente iperbolica:

$$\phi(u) = \tanh(u) = \frac{e^{u} - e^{-u}}{e^{u} + e^{-u}} \tag{5}$$

Esistono moltissime altre funzioni deputate a tale scopo, e la scelta di quale utilizzare dipende strettamente dall'applicazione finale.
I singoli neuroni di una rete sono organizzati in livelli. In particolare, uno *strato* (Figura 6) è un gruppo di neuroni che condividono gli ingressi. Uno strato di dimensione S è uno strato dotato di S neuroni e di conseguenza di S uscite.

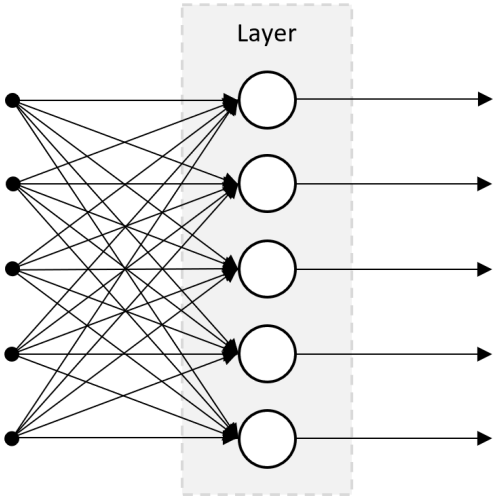

Figura 6: Strato di neuroni.

Una serie di strati costituisce la rete neurale vera e propria (Figura 7). Si possono distinguere tre sezioni principali:

- lo strato di ingresso (*input layer*), di dimensione pari al numero di input della rete;
- lo strato di uscita (*output layer*), di dimensione pari al numero di output della rete;
- uno o più strati nascosti (*hidden layers*), non direttamente accessibili dall'esterno e tipicamente di dimensione maggiore rispetto agli strati di ingresso e uscita.

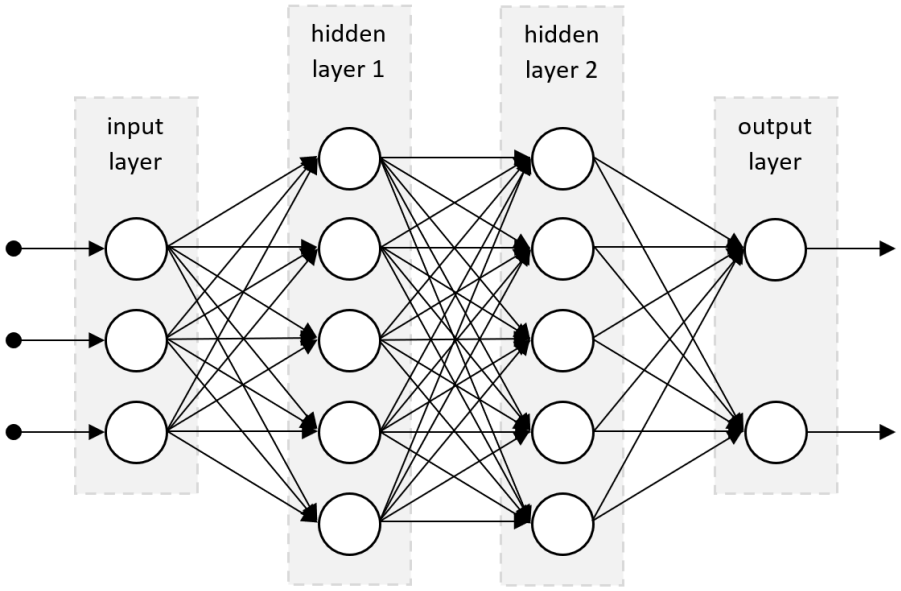

Figura 7: Rete neurale artificiale.

Le uscite di tutti i neuroni di ciascuno strato costituiscono gli ingressi dello strato successivo e ognuno di questi collegamenti è contraddistinto da un peso. Incrementando il numero di strati nascosti e/o la loro dimensione, aumenta la complessità sia della rete sia dei problemi che essa è in grado di risolvere.

2.2 ALLENAMENTO

Una rete neurale è potenzialmente in grado di risolvere un'ampia gamma di problemi diversi, ma non prima di averne impostato i pesi in maniera opportuna. Questo processo di variazione dei parametri è universalmente noto con il nome di *training* (traducibile in italiano con *allenamento* o *addestramento*). Ogni aggiustamento dei pesi avviene unicamente sulla base di stimoli esterni e va a modificare il comportamento del modello. Per mezzo di un corretto allenamento, dunque, una rete neurale artificiale è in grado di apprendere come assolvere ad un dato compito.
Esistono tre principali paradigmi di apprendimento, legati ad altrettante modalità di allenamento della rete:

- *apprendimento supervisionato*: il sistema viene allenato sfuttando un dataset di possibili dati di input combinato con il set dei corrispondenti output desiderati. In altre parole, si forniscono

al modello degli esempi sulla reazione che dovrebbe avere in risposta a certi stimoli. L'obiettivo è quello di produrre una funzione che approssimi nel miglior modo possibile la relazione che intercorre tra i dati di ingresso e quelli di uscita. In questo modo, la rete sarà in grado di generalizzare, ovvero di rispondere nella maniera appropriata anche a stimoli diversi rispetto a quelli utilizzati per addestrarla;

- *apprendimento non supervisionato*: durante l'addestramento sono forniti alla rete esclusivamente i dati di ingresso, senza alcuna indicazione sulle uscite desiderate. È il modello che, autonomamente e nella maniera che ritiene più adeguata, classifica ed organizza i dati sulla base di caratteristiche comuni. Questo paradigma di apprendimento trova applicazione, ad esempio, per lo sviluppo di sistemi di filtraggio o di compressione dei dati;

- *apprendimento per rinforzo*: mira a realizzare degli agenti autonomi che prendano delle decisioni allo scopo di massimizzare una certa funzione "premio" in un certo orizzonte temporale.

In questo lavoro ci si focalizzerà sull'apprendimento supervisionato. Tipicamente, i problemi risolvibili da una rete allenata in questo modo sono di due tipologie:

- *regressione*, ovvero la stima di una relazione funzionale che correli i dati di output desiderati, di tipo numerico, con quelli di input forniti. Esempi di regressione sono: (*i*) la valutazione del prezzo di una casa, a partire da una insieme di parametri forniti in ingresso quali metratura, locazione, condizioni dell'edificio, ecc. [31], o (*ii*) la stima dell'efficienza del carburante in mezzi a motore noti peso, potenza, cilindrata, ecc. [37];

- *classificazione*, che consiste nell'identificazione, da parte del sistema automatico, della "categoria" (*classe*) di appartenenza di un certo oggetto, note le informazioni su alcune sue caratteristiche. A differenza dei regressori, l'output di un classificatore è di tipo categorico, cioè definito in un dominio discreto avente un numero finito di elementi. Un esempio di classificazione è il riconoscimento da un immagine di caratteri scritti a mano. In questo caso, le possibili classi sono i vari simboli alfanumerici

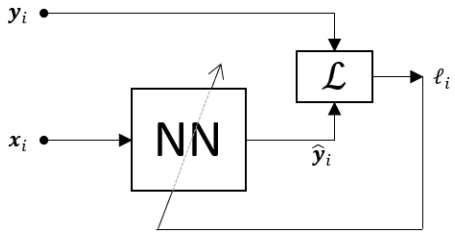

Figura 8: Apprendimento supervisionato.

("A", "B", "C", ..., "1", "2", ...), mentre l'input è costituito dall'immagine stessa (una matrice di pixel ognuno caratterizzato da una terna di valori numerici RGB o HSB).

Lo schema di principio dell'apprendimento supervisionato è esposto in Figura 8. Sia $\boldsymbol{x}_i \in \mathbb{R}^m$ il vettore degli m ingressi alla rete e sia $\boldsymbol{y}_i \in \mathbb{R}^n$ il vettore delle n uscite. Il dataset di allenamento è pertanto composto da una serie di B campioni $\{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_B\}$ che rappresentano alcuni possibili ingressi, ai quali corrispondono le B uscite desiderate $\{\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_B\}$. L'$i$-esimo passo del processo di addestramento consiste nella stimolazione della rete neurale (indicata in figura con "NN") con l'ingresso $\boldsymbol{x}_i$, al quale consegue l'uscita $\hat{\boldsymbol{y}}_i$. Il confronto tra quest'ultimo valore e quello richiesto $\boldsymbol{y}_i$ permette di ricavare un segnale di errore $\ell_i \in \mathbb{R}$ che indichi l'accuratezza dell'attuale stima. A questo punto, il modello modifica i pesi delle connessioni in funzione di ℓ_i.
La variazione dei parametri segue una logica di ottimizzazione: l'obiettivo è trovare la configurazione che minimizza una certa *funzione di perdita* (*loss function*) $\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)$ su tutti gli esempi forniti nel dataset di allenamento:

$$\hat{\boldsymbol{w}} = \underset{\boldsymbol{w}}{\operatorname{argmin}} \left(\sum_{i=1}^{B} \mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) \right) \tag{6}$$

(dove con $\boldsymbol{w}$ si è indicato in maniera generica un vettore che include tutti i pesi della rete). La scelta del tipo di funzione di perdita da utilizzare è strettamente legata all'applicazione. In compiti di regressione, per esempio, nei quali è richiesto alla rete di elaborare una funzione approssimante di un qualche fenomeno, la qualità di una stima è legata alla sua "vicinanza" al valore vero. In tali circostanze risulta pertanto utile calcolare una "distanza media" tra $\boldsymbol{y}_i$ e $\hat{\boldsymbol{y}}_i$:

$$\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \|\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i\|. \tag{7}$$

Questo approccio non è valido per attività di classificazione, perchè non esiste alcun concetto di distanza per quanto riguarda le classi (a

meno che non sia definito a priori), anche se a queste occorre spesso assegnare un indice numerico e di conseguenza un ordine. Tornando all'esempio precedente, un classificatore addestrato per riconoscere lettere o cifre da un'immagine non dovrebbe correlare l'accuratezza della propria stima alla vicinanza con il valore vero, poichè se l'uscita desiderata è "3" allora un output "4" non è migliore di un "8". Una funzione di perdita adeguata allo scopo è:

$$\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \begin{cases} 0 & \text{se } \boldsymbol{y}_i = \hat{\boldsymbol{y}}_i \\ 1 & \text{altrimenti} \end{cases} \tag{8}$$

che elimina il problema dell'ordinalità delle classi.
Molti classificatori, tuttavia, sono addestrabili per fornire in uscita una distribuzione di probabilità di appartenenza a tutte le classi. Se N_C è il numero di possibili categorie, allora un modello di questo tipo è dotato esattamente di N_C uscite, dove il c-esimo output designa la probabilità che l'oggetto in questione sia assegnabile alla classe c-esima. In queste circostanze si preferisce sfruttare la cosidetta *entropia incrociata*:

$$\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = -\sum_{c=1}^{N_C} \boldsymbol{y}_{i,c} \log\left(\hat{\boldsymbol{y}}_{i,c}\right) \tag{9}$$

(dove $\boldsymbol{y}_{i,c}$ è la c-esima componente del vettore di uscita $\boldsymbol{y}$). Se $\boldsymbol{y}_i$ segue la codifica *one-hot*, ovvero contiene $n-1$ zeri ed un singolo 1 in posizione c (la classe corretta), allora $\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i)$ si riduce semplicemente a $-\log\left(\hat{\boldsymbol{y}}_{i,c}\right)$, dove $\hat{\boldsymbol{y}}_{i,c}$ è la probabilità stimata dal classificatore per la classe corretta (c). In questo modo, il calcolo dell'entropia incrociata restituisce valori molto alti se $\hat{\boldsymbol{y}}_{i,c}$ è molto bassa (tendenti ad infinito per $\hat{\boldsymbol{y}}_{i,c} \to 0$), mentre ritorna 0 (il minimo) nel caso la probabilità stimata per la classe corretta sia il 100%.
Tipicamente, il dataset impiegato in un allenamento supervisionato non viene fornito come ingresso alla rete una singola volta. Al contrario, questo ciclo (che viene chiamato *epoca*) viene ripetuto una pluralità di volte (numero di epoche). La gran mole di dati necessari, inoltre, rende necessaria una loro suddivisione in lotti (*batches*), per via dei limiti imposti in termini di memoria e di potenza di calcolo dagli elaboratori, i quali non sarebbero in grado di eseguire l'intero processo in un solo passo. Si definisce la *batch size* come il numero di campioni di esempio forniti alla rete per ogni passo di addestramento. Un alto batch size consente di ridurre i tempi necessari per completare il procedimento, ma aumenta il rischio di errori del tipo *out of memory*.

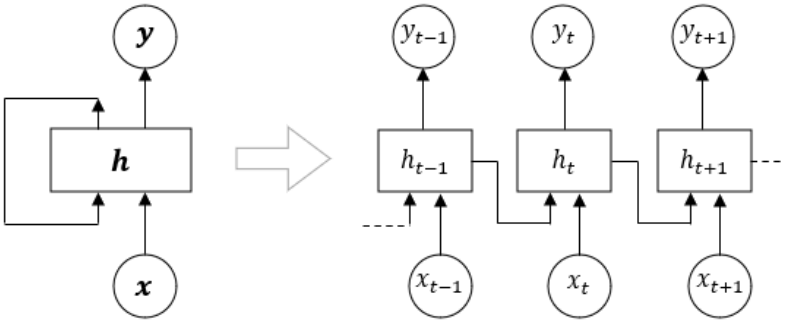

Figura 9: Rete neurale ricorrente.

Generalmente la fase di training è piuttosto lunga e laboriosa, poiché richiede, ad ogni passo, il calcolo del gradiente della funzione di perdita al fine di ricercarne il minimo. Inoltre, è necessario imporre dei vincoli sul dataset di addestramento, che deve essere abbastanza esteso oltre che rappresentativo di tutti i possibili ingressi, in modo da dotare la rete di un'esperienza tale da permetterle di generalizzare correttamente.

2.3 TIPOLOGIE DI RETI

Nella struttura di rete neurale presentata in precedenza, il flusso delle informazioni è unidirezionale: lo strato di ingresso attiva il primo strato nascosto che a sua volta ne attiva altri fino ad arrivare allo strato di uscita. Questa tipologia di rete, nota in letteratura come *rete neurale feed-forward*, non ha alcun tipo di memoria relativo agli ingressi o alle uscite degli istanti passati. Perciò, risulta adeguata a risolvere solamente problemi in cui la variabile temporale non è influente, come ad esempio la regressione di funzioni statiche. Per compiti di natura sequenziale, in cui è necessario che ogni output sia correlato ai precedenti (ad esempio la generazione di traiettorie nello spazio o di frasi in lingua italiana) conviene affidarsi alle *reti neurali ricorrenti* (*recurrent neural network*, RNN). In queste strutture (Figura 9), l'uscita di ogni nodo elementare al tempo t ($\boldsymbol{y}_t$) non dipende unicamente dagli ingressi in tale istante ($\boldsymbol{x}_t$), ma anche dal suo *stato* ($\boldsymbol{h}_t$), il quale viene continuamente aggiornato. Un neurone di questo tipo (Figura 10) è modellabile nella seguente maniera (tralasciando i bias):

$$\begin{cases} \boldsymbol{h}_t = \phi_h \left(\boldsymbol{w}_h \boldsymbol{x}_t + \boldsymbol{g}_h \boldsymbol{h}_{t-1} \right) \\ \boldsymbol{y}_t = \phi_y \left(\boldsymbol{w}_y \boldsymbol{h}_t \right) \end{cases} \tag{10}$$

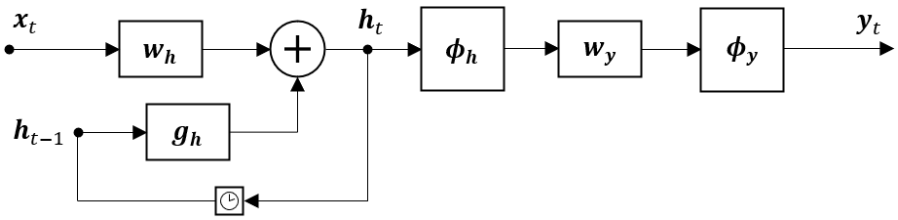

Figura 10: Nodo di una rete neurale ricorrente.

dove w_h, w_y, g_h indicano dei pesi e ϕ_h, ϕ_y sono funzioni di attivazione. Tale configurazione dona alla rete una sorta di memoria, permettendole di elaborare ingressi forniti sotto forma di successione anzichè di singolo campione. Quest'abilità rende le RNN perfettamente idonee alla risoluzione di problemi sequenziali, tra cui la predizione.
Un'altra architettura interessante è offerta dalle cosidette *reti neurali convoluzionali*, le quali sfruttano, per l'appunto, il concetto di convoluzione. Il prodotto di convoluzione di una matrice $A \in \mathbb{R}^{R \times C}$ con un "filtro" (*kernel*) $K \in \mathbb{R}^{D \times D}$ ($D \leq R$, $D \leq C$) restituisce una matrice B tale che:

$$B_{x,y} = \sum_{j=-b}^{+b} \sum_{i=-a}^{+a} K_{i,j} A_{x-j,y-i} \tag{11}$$

(dove le righe e le colonne di K sono indicizzate rispettivamente da $-a$ a a e da $-b$ a b). Un esempio di convoluzione è riportato in Figura 11.

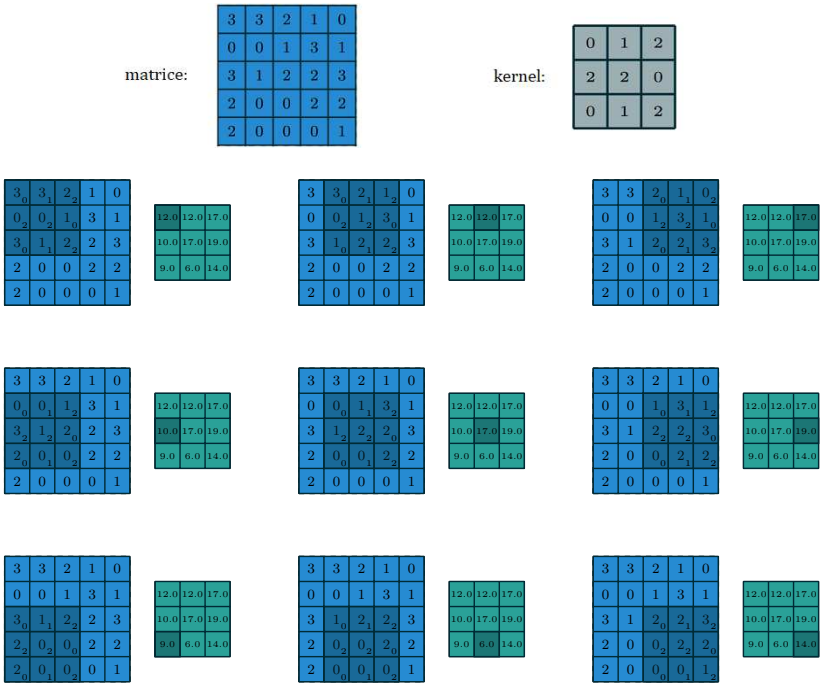

Figura 11: Esempio di convoluzione 2D tra matrici [7].

Come si può notare, l'elemento di B in posizione (x, y) non dipende solo dall'elemento di A nella stessa posizione, ma anche dai valori degli elementi adiacenti pesati dalla matrice K. Incrementando D, aumenta anche il numero di "vicini" considerati.
Questo aspetto rende le reti convoluzionali molto efficienti nell'elaborazione di ingressi che rappresentano dei dati spaziali in due dimensioni, in cui il concetto di vicinanza è influente (ad esempio le immagini). Sono largamente utilizzate nelle attività di riconoscimento di oggetti e di visione artificiale, per via delle loro prestazioni nettamente superiori rispetto a reti neurali feed-forward di pari complessità.
Il prodotto di convoluzione può essere definito in maniera analoga anche in uno spazio a dimensione n maggiore di 2. In questo caso l'operazione viene svolta su tensori[1] n-dimensionali, anziché su matrici. In Figura 12 viene riportato lo schema di principio della convoluzione in 3D.

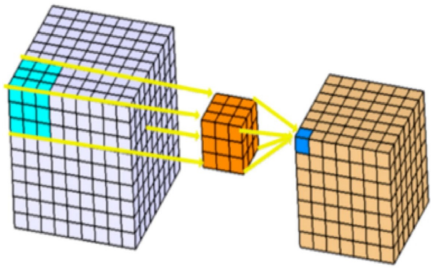

Figura 12: Principio della convoluzione 3D [34].

Combinando le proprietà delle reti ricorrenti e di quelle convoluzionali, si ottengono le *reti neurali ricorrenti convoluzionali* (*ConvRNN*) [49, 44]. Una cella di una ConvRNN opera in maniera simile ad una classica RNN, con le seguenti eccezioni:

- i prodotti scalari sono sostituiti da convoluzioni (indicate nel seguito con il simbolo "$*$");
- gli ingressi ($\boldsymbol{X} \in \mathbb{R}^{R \times C}$), l'uscita ($\boldsymbol{Y} \in \mathbb{R}^{R \times C}$) e lo stato ($\boldsymbol{H} \in \mathbb{R}^{R \times C}$) sono delle matrici (nel caso 2D) le cui dimensioni hanno un significato spaziale (pixel, coordinate xy, ...).

1 I tensori costituiscono una generalizzazione delle matrici al caso multi-dimensionale. I vettori sono tensori unidimensionali, mentre le matrici sono tensori bidimensionali.

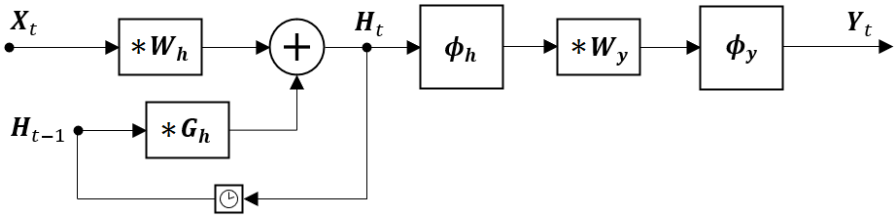

Figura 13: Nodo di una rete neurale ricorrente convoluzionale.

Il modello del nodo elementare di una rete di questo tipo (Figura 13) ricalca l'Equazione (10):

$$\begin{cases} \boldsymbol{H}_t = \phi_h \left(\boldsymbol{W}_h * \boldsymbol{X}_t + \boldsymbol{g}_h * \boldsymbol{H}_{t-1} \right) \\ \boldsymbol{Y}_t = \phi_y \left(\boldsymbol{W}_y * \boldsymbol{H}_t \right). \end{cases} \tag{12}$$

Il principale vantaggio nell'utilizzo di questi modelli è la loro attitudine a gestire ingressi sotto forma di sequenze spazio-temporali, quali ad esempio video o traiettorie. Sono impiegati con successo in diverse applicazioni, quali le previsioni del tempo e l'anticipazione dei flussi del traffico stradale.

2.4 L'architettura sequence-to-sequence

L'architettura *Sequence-to-sequence* [38] (spesso abbreviata con *Seq2seq*) è stata proposta nel 2014 da alcuni ricercatori di Google per la risoluzione di problemi di traduzione di frasi da una lingua (l'inglese, nella loro pubblicazione) ad un'altra (il francese). Negli anni seguenti, le potenzialità di questa struttura sono state provate in molti altri ambiti di differente natura, come il calcolo [43] e la predizione [15].
Sebbene le classiche RNN siano adeguate nell'elaborazione di sequenze (come lo sono le frasi), un loro limite risiede nella corrispondenza uno a uno tra ingressi ed uscite: ad ogni input (al tempo t) consegue un certo output (sempre al tempo t). Questo approccio non è adeguato allo svolgimento di compiti di traduzione, poiché già per determinare la prima parola corretta nella nuova lingua è dapprima necessario comprendere il contesto della frase espressa in lingua originale.
L'idea del metodo Seq2seq è quella di trasformare una sequenza in input di lunghezza arbitraria ($x_{1:m} = \{x_1, ..., x_m\}$) in un vettore "nascosto" di lunghezza fissa ($\boldsymbol{h}$), il quale viene poi convertito in un'altra sequenza di lunghezza anche differente ($y_{1:n} = \{y_1, ..., y_n\}$).
Tornando all'esempio della traduzione, il vettore nascosto $\boldsymbol{h}_t$ contie-

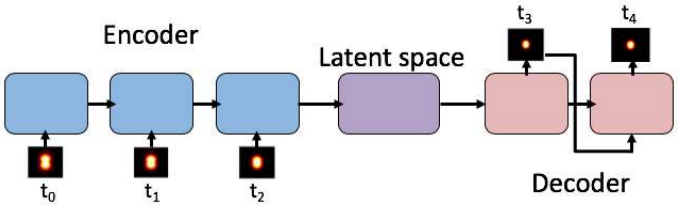

Figura 14: Architettura Sequence-to-sequence [15].

ne una sorta di rappresentazione del significato della frase fino alla t-esima parola. Se la frase di input contiene m parole, la rappresentazione del contesto, ovvero del significato dell'intera frase, è data dal valore finale del vettore nascosto ($\boldsymbol{h}_m$).
I componenti fondamentali di quest'architettura (Figura 14) sono:

- l'*encoder*, tipicamente una RNN, che computa il contesto in funzione della sequenza in ingresso;
- il *decoder*, un'altra RNN, che ha il compito di determinare la corretta sequenza di uscita a partire dal contesto.

Per avere una corretta rappresentazione dell'intera sequenza in ingresso, lo stato del decoder viene inizializzato con lo stato finale dell'encoder.
Per attività di traduzione o, in generale, di generazione di testi, il decoder è solitamente allenato in modo da fornire una molteplicità di possibili output, ognuno caratterizzato da una certa probabilità (comunemente viene utilizzata l'entropia incrociata come funzione di perdita). Questo aspetto consente ai traduttori, ad esempio, di generare più possibili traduzioni per la stessa frase di input.
Il problema da risolvere a questo punto è quello di generare le F frasi più probabili a partire da delle distribuzioni di probabilità sulle V parole presenti nel vocabolario della lingua di destinazione. Dato che V è un numero molto grande (in [4] è indicato da 10000 a 100000), è impensabile confrontare tutte le V^P possibili combinazioni di P parole in cerca di quella migliore (dove, oltretutto, P non è noto a priori). Occorre pertanto affidarsi a metodi alternativi alla cosidetta *ricerca esaustiva*.
L'algoritmo più immediato è la *greedy search* (letteralmente "ricerca avida"), che consiste nella selezione ad ogni passo di traduzione della parola del vocabolario più probabile. Il limite di questo approccio è la sua incapacità di generare più di una frase (vincolo $F = 1$). Inoltre,

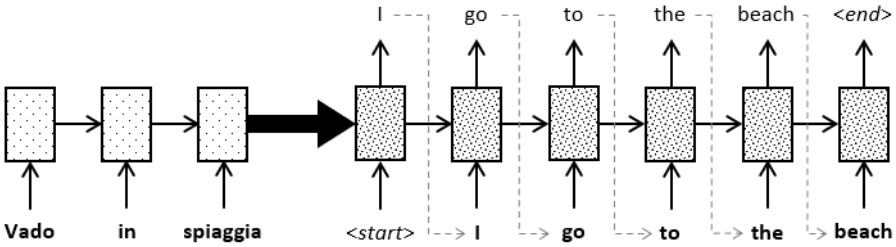

Figura 15: Traduzione italiano-inglese tramite Seq2seq.

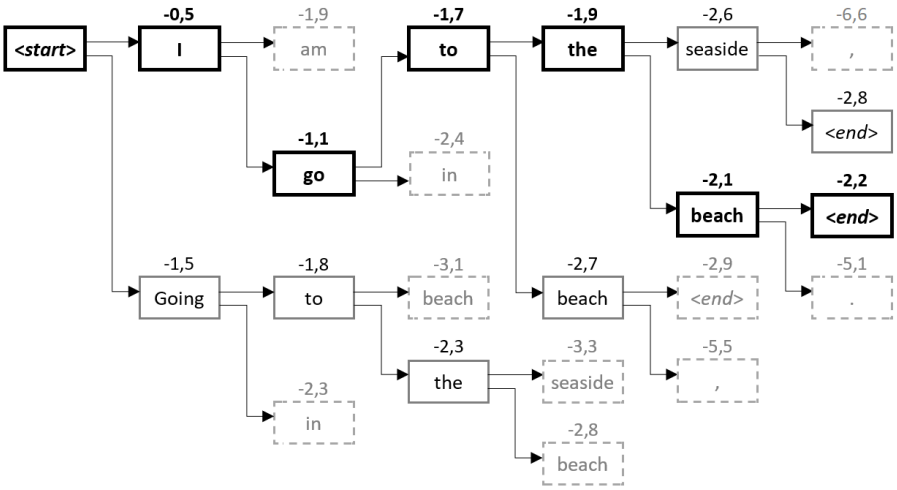

Figura 16: Beam search applicata alla traduzione.

per sua natura, accade spesso che la frase risultante non sia quella effettivamente con probabilità maggiore, poiché nel procedimento vengono ignorate molte combinazioni interessanti.
Una soluzione a questo limite consiste nel tenere traccia, per ogni passo del decoder, delle F sequenze parziali più probabili. In questo modo, alla fine del processo si ottengono le migliori F sequenze complete. Questo metodo prende il nome di *beam search* [45]. Si supponga di dover tradurre la frase "vado al mare" dall'italiano all'inglese, in modo da ottenere "I go to the beach" (Figura 15). Il procedimento (con $F = 2$) è schematizzato in Figura 16, dove i numeri (z) indicano la probabilità logaritmica delle sequenze parziali (Prob $= e^z$). Come si vede, ad ogni passo del decoder, le prime $F = 2$ sequenze in ordine di probabilità sono mantenute, mentre le altre scartate. Il risultato finale è la coppia di frasi:

- "I go to the beach", con probabilità $e^{-2.2} \approx 11\%$;
- "I go to the seaside", con probabilità $e^{-2.8} \approx 6\%$.

Applicando una greedy search, che di fatto equivale ad una beam

search di parametro $F = 1$, si otterrebbe solamente la prima possibilità.
Occorre precisare che la beam search non garantisce il conseguimento del risultato ottimale. Tuttavia, una scelta ponderata del parametro F consente comunque di ottenere esiti soddisfacenti, con un sensibile incremento di efficienza rispetto ad una laboriosa ricerca esaustiva.

3

SISTEMA DI PREDIZIONE

Questo capitolo descrive nel dettaglio l'algoritmo di anticipazione proposto in questa tesi. Nella prima sezione, viene fornita una panoramica del sistema complessivo e del suo principio di funzionamento. Successivamente, sono illustrati tutti i passaggi logici che portano al risultato finale, cioè la predizione dei movimenti futuri di un essere umano.
Il software è sviluppato con il linguaggio Python. In particolare, per la gestione degli array si ricorre alla libreria NumPy [14], per la conversione delle rotazioni si sfrutta SciPy [41], Scikit-learn [30] fornisce supporto per la riduzione della dimensionalità dei dati (vedi Sezione 3.3), mentre l'implementazione delle reti neurali avviene per mezzo di TensorFlow [1].

3.1 PANORAMICA

Nel suo complesso, il sistema di predizione implementato può essere schematizzato come in Figura 17. A partire da un campione osservato di movimenti ($\boldsymbol{P}_{1:T_o}$), l'obiettivo è generare una molteplicità di pose future plausibili ($\hat{\boldsymbol{P}}_{T_o+1:T_s}$), che stimino nella maniera più accurata possibile il movimento futuro reale $\boldsymbol{P}_{T_o+1:T_s}$. L'idea di base è di considerare ogni singola posa $\boldsymbol{P}_t$ come un insieme di punti $\boldsymbol{X}_t$, trattando quindi un movimento umano $\boldsymbol{P}_{1:T}$ come un set di traiettorie $\boldsymbol{X}_{1:T}$.
Come si vede nell'immagine, l'input $\boldsymbol{P}_{1:T_o} \in \mathbb{R}^N$ viene scisso in $J + 2 = 20$ traiettorie: una ($\boldsymbol{X}_{T,1:T_o} \in \mathbb{R}^3$) per la traslazione globale del corpo (in particolare del root keypoint), una ($\boldsymbol{X}_{0,1:T_o} \in \mathbb{R}^{N_{pr}}$) per l'orientazione di quest'ultimo e altre J per le rotazioni dei giunti ($\boldsymbol{X}_{j,1:T_o} \in \mathbb{R}^{N_{pr}}$, dove j indica l'indice del giunto, con riferimento alla Figura 2). Con la rappresentazione asse-angolo introdotta in Sezione 1.3 si ha $N_{pr} = 3$, il che rende tridimensionali tutte le traiettorie. Esse, tuttavia, non sono tutte definite nello stesso spazio. Le coordinate della traslazione globale ($\boldsymbol{X}_t$) variano senza vincoli entro $\mathbb{R}^3$, perché sono effettivamente riferite allo spazio fisico 3D. I parametri di ognuna

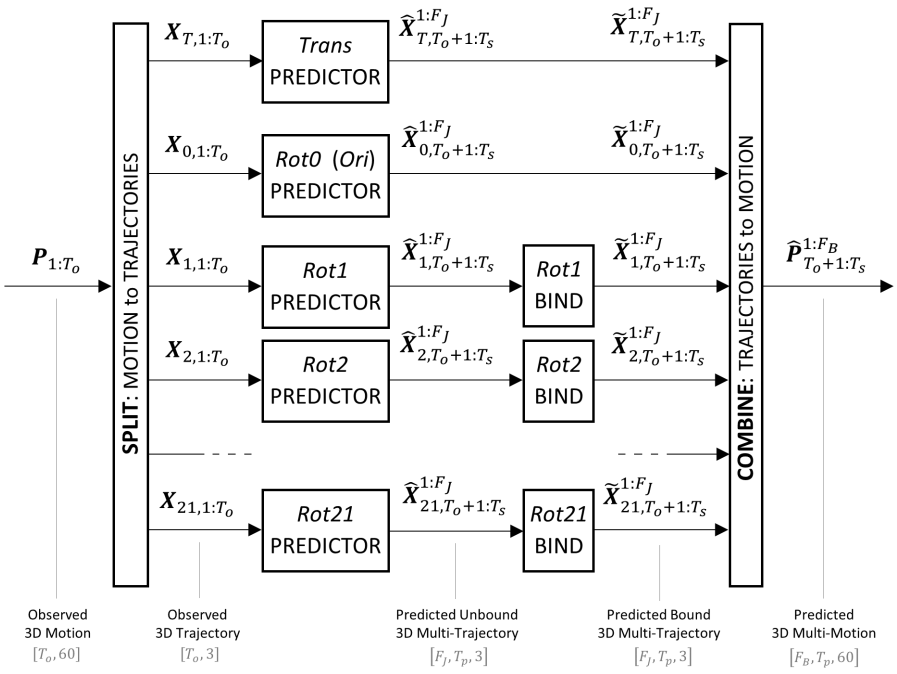

Figura 17: Schema a blocchi del sistema complessivo.

delle $J + 1$ rotazioni, invece, sono definiti nello spazio della notazione asse-angolo, il quale è tridimensionale ma limitato entro la sfera centrata nell'origine di raggio 2π.

A valle della scomposizione dei movimenti in traiettorie, viene operata la predizione (multi-futuro) vera e propria, che avviene in maniera indipendente per ognuno dei giunti. L'output di questo stadio è, per ciascuna delle $J + 2$ terne, un set di F_J traiettorie future plausibili ($\hat{\boldsymbol{X}}^{1:F_J}_{T_o+1:T_s}$, con $\hat{\boldsymbol{X}}^f_t \in \mathbb{R}^{F_J \times 3}$). Ulteriori dettagli sono forniti nel seguito.

La fase successiva (spiegata in Sezione 3.4) è dedicata all'applicazione dei vincoli anatomici alle traiettorie $\hat{\boldsymbol{X}}^{1:F_J}_{T_o+1:T_s}$, in modo da ottenere delle rotazioni di giunto anatomicamente plausibili per ogni istante dell'orizzonte predetto ($\tilde{\boldsymbol{X}}^{1:F_J}_{T_o+1:T_s}$, con $\tilde{\boldsymbol{X}}^f_t \in \mathbb{R}^{F_J \times 3}$). Dal procedimento sono escluse le traiettorie legate al root keypoint, perché posizione e orientazione globale del corpo non risentono di questo tipo di vincoli (eventuali vincoli legati all'ambiente circostante non sono considerati in questa tesi).

Le singole traiettorie predette e vincolate sono infine combinate al fine di ottenere F_B movimenti umani futuri plausibili ($\hat{\boldsymbol{P}}^{1:F_B}_{T_o+1:T_s}$, con $\hat{\boldsymbol{P}}^f_t \in \mathbb{R}^{F_B \times N}$) che non violino i vicoli anatomici.

Il problema di predizione di movimentazioni umane è stato composto in vari sotto-problemi di predizione di traiettorie. Occorre a questo punto sviluppare un algoritmo adatto a svolgere quest'ultimo compi-

to. Il metodo qui utilizzato prende ispirazione da un'architettura già esistente, denominata *Multiverse*, che verrà discussa nel dettaglio nella Sezione 3.2. Multiverse, che opera con dati derivanti da scene video, si occupa di generare una predizione multi-futuro delle traiettorie percorse da un individuo all'interno di una scena in due dimensioni. Si tracciano in particolare le posizioni 2D del "centroide" del soggetto considerato, che viene quindi trattato come un semplice punto. È possibile allenare un modello di questo tipo in modo che fornisca le stesse informazioni riguardo ad un giunto/keypoint, piuttosto che ad una persona. Le traiettorie dei giunti sono riferite in uno spazio diverso, e a dimensione maggiore, rispetto a quelle richieste in input da Multiverse. Ciò comporta la necessità di inserire un blocco "TRSF" che trasformi i dati 3D iniziali ($\boldsymbol{X}_{j,t} \in \mathbb{R}^3$), portandoli in uno spazio interpretabile da Multiverse ($\boldsymbol{U}_{j,t} \in \mathbb{R}^2$). Quest'ultimo sarà quindi in grado di fornire una predizione multi-futuro ($\hat{\boldsymbol{U}}^{1:F_J}_{T_o+1:T_s}$), la quale verrà riconvertita nello spazio iniziale ($\hat{\boldsymbol{X}}^{1:F_J}_{T_o+1:T_s}$) dal blocco "TRSF^{-1}". Lo schema a blocchi del processo è riassunto in Figura 18 (dove "MV" indica un modello Multiverse).

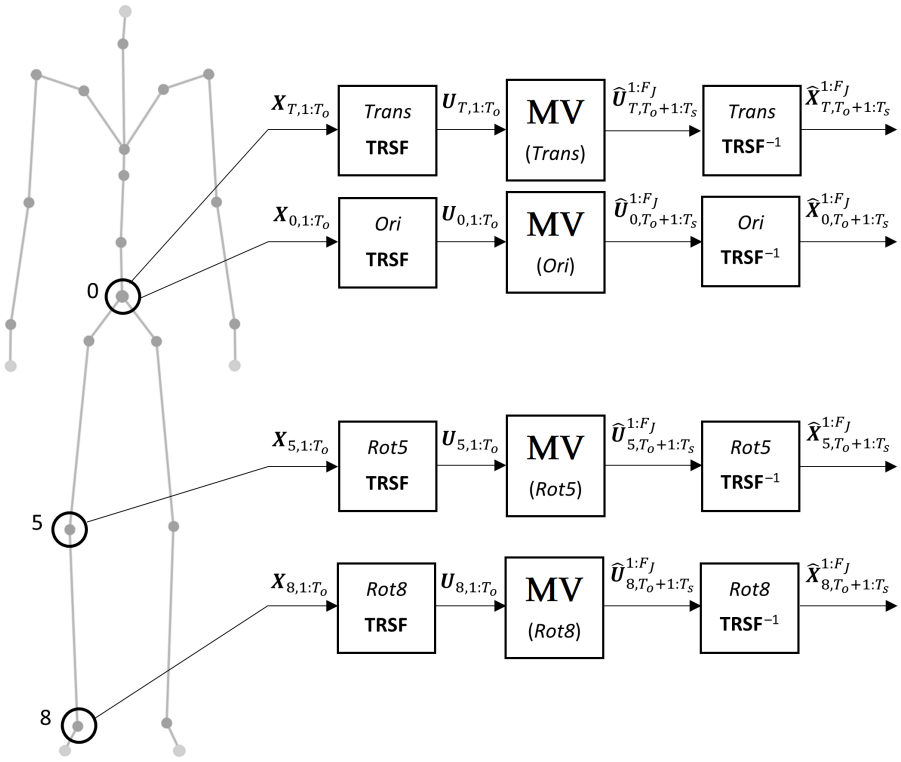

Figura 18: Predizione delle traiettorie di giunto (schema a blocchi).

Il modello Multiverse [20] è stato implementato al fine di fornire una predizione multi-futuro della traiettoria percorsa da una persona all'interno di una scena bidimensionale, tipicamente video. Multiverse si presta bene a svariate applicazioni, quali la guida autonoma, il tracciamento di oggetti a lungo termine e la pianificazione del moto robotico.
Semplificando, il modello riceve in ingresso una traiettoria ($\boldsymbol{U}_{1:T_o} = \{\boldsymbol{U}_1, ..., \boldsymbol{U}_{T_o}\}$), intesa come successione temporale di T_o vettori 2D indicanti la posizione dell'individuo considerato, denominato *agente controllato*, ad ogni istante di osservazione ($\boldsymbol{U}_t \in \mathbb{R}^2$). Le coordinate delle traiettorie sono espresse nello spazio dei pixel ($0 \leq \boldsymbol{U}_{t,x} \leq V_W$, $0 \leq \boldsymbol{U}_{t,y} \leq V_H$), essendo Multiverse progettato per elaborare singoli fotogrammi da scene video (aventi risoluzione $V_H \times V_W$). A partire da questi dati, il sistema è in grado di fornire in uscita un certo numero (F_J) di traiettorie future plausibili per l'agente controllato ($\hat{\boldsymbol{U}}^{1:F_J}_{T_o+1:T_s} = \{\hat{\boldsymbol{U}}^{1:F_J}_{T_o+1}, ..., \hat{\boldsymbol{U}}^{1:F_J}_{T_s}\}$), ciascuna di lunghezza T_p impostabile liberamente. Ad ogni scenario futuro f è associata una certa probabilità φ^f, calcolata internamente dal modello. Il funzionamento di Multiverse è schematizzato in Figura 19.

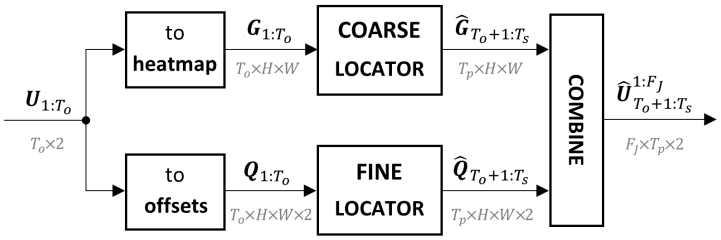

Figura 19: Schema a blocchi del modello Multiverse.

Il modello è costituito da due sezioni fondamentali: il *Coarse Locator* (CL) e il *Fine Locator* (FL), che consistono in due distinte architetture Sequence-to-sequence. La prima ha il compito di determinare una distribuzione di probabilità delle possibili destinazioni dell'agente controllato nel piano 2D di riferimento (quello dei video frames). La seconda, invece, affina la predizione del Coarse Locator, ricavando in maniera più precisa l'esatta posizione del soggetto.
Entrambe le reti (CL e FL) non operano direttamente con la traiettoria di input ($\boldsymbol{U}_{1:T_o}$), bensì sfruttano una sua codifica. I blocchi "to heatmap" e "to offsets" di Figura 19 convertono l'ingresso nella rappresentazione

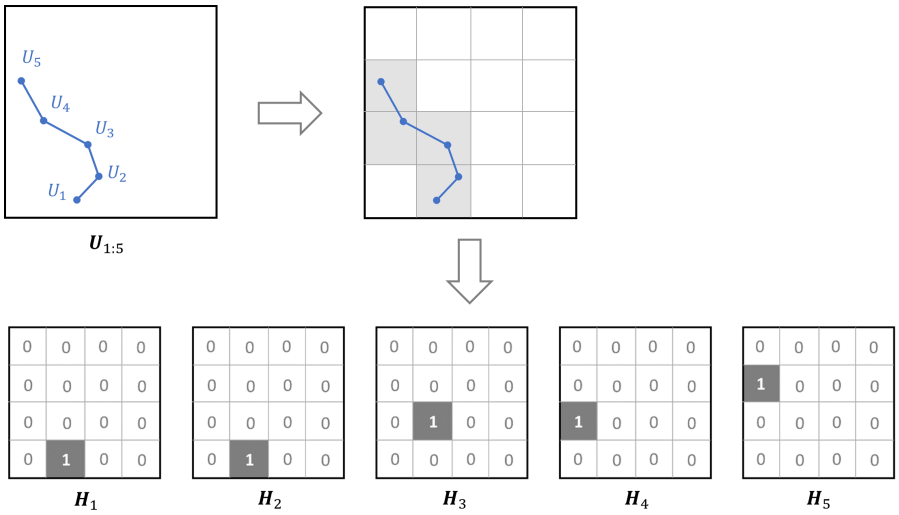

Figura 20: Multiverse: codifica di una traiettoria all'interno della griglia 2D.

richiesta rispettivamente dal Coarse e dal Fine Locator. Il principio del blocco "to heatmap" è schematizzato in Figura 20: ogni punto $\boldsymbol{U}_t$ della traiettoria viene codificato con una matrice $\boldsymbol{G}_t \in \mathbb{R}^{H \times W}$ tale che:

$$\boldsymbol{G}_t^{(i,j)} = \begin{cases} 1 & \text{se } \boldsymbol{U}_t \in c^{(i,j)} \\ 0 & \text{altrimenti} \end{cases} \tag{13}$$

dove $c^{(i,j)}$ è la cella alla posizione (i, j) di una griglia di dimensioni $H \times W$ che include tutta la scena 2D. Nel suo cammino, l'agente controllato attraversa un certo numero di celle $c^{(i,j)}$; questa informazione viene esplicitamente elaborata entro il Coarse Locator.

Un'ulteriore trasformazione dei dati di input è attuata dal blocco "to offsets" di Figura 19. In questo caso, per ogni punto della traiettoria viene calcolato un offset 2D rispetto ai centri di tutte le $H \cdot W$ celle della griglia (come in Figura 21, dove è riportato il procedimento per la cella $c^{(1,3)}$). In questo modo, per ciascuna cella della griglia 2D si ha un'indicazione riguardo la relativa distanza del soggetto. Ogni punto $\boldsymbol{U}_t$ viene quindi codificato con un tensore $\boldsymbol{Q}_t \in \mathbb{R}^{H \times W \times 2}$ tale che:

$$\boldsymbol{Q}_t^{(i,j)} = \boldsymbol{U}_t - \boldsymbol{C}^{(i,j)} \tag{14}$$

dove $\boldsymbol{C}^{(i,j)}$ è il centro della cella $c^{(i,j)}$.

L'uscita del blocco "to heatmap" ($\boldsymbol{G}_{1:T_o}$) costituisce l'ingresso del Coarse Locator (letteralmente *localizzatore grossolano*). Il CL è una struttura Seq2seq, in cui le funzioni dell'encoder e del decoder sono svolte da due reti neurali ricorrenti convoluzionali [49, 44], scelta che dona al modello una maggior efficienza nell'apprendimento delle dipendenze

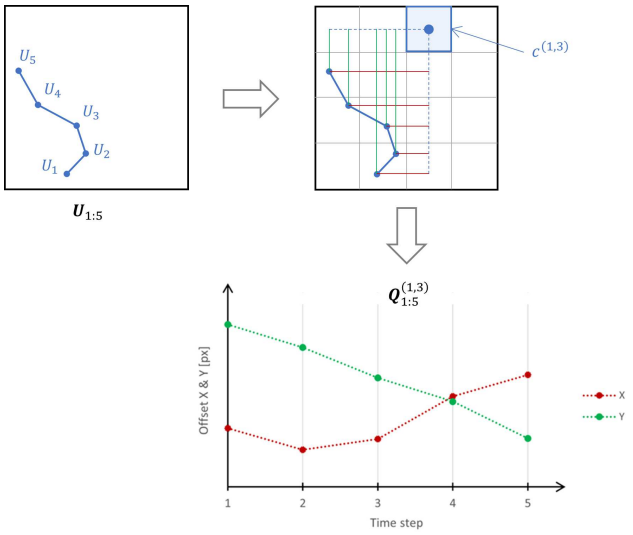

Figura 21: Multiverse: determinazione degli offset 2D di una traiettoria.

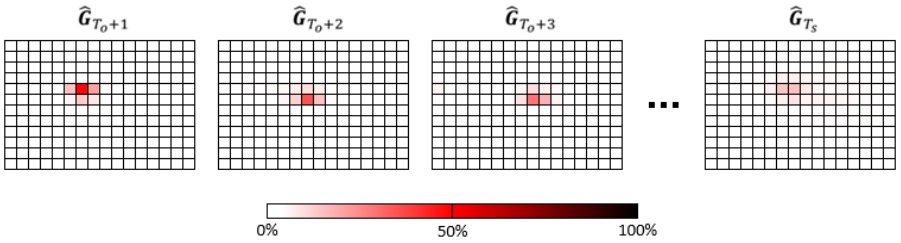

Figura 22: Multiverse: heatmap generabili dal Coarse Locator.

spazio-temporali. Encoder e decoder sono dotati di uno strato nascosto di dimensioni rispettivamente S_{enc} e S_{dec}, mentre lo stato latente è dato da una matrice $H \times W \times d$. A partire dalla sequenza $\boldsymbol{G}_{1:T_o}$, il Coarse Locator si occupa di generare una sequenza $\hat{\boldsymbol{G}}_{T_o+1:T_s}$ contenente T_p heatmap $\hat{\boldsymbol{G}}_t \in \mathbb{R}^{H \times W}$, ognuna associata ad un istante temporale della sequenza di uscita (Figura 22). Ciascuna cella della t-esima heatmap contiene un valore decimale che identifica con quale probabilità (secondo il modello) l'agente controllato si troverà in tale locazione al t-esimo istante di predizione. Per svolgere tale compito, il CL risolve un problema di classificazione, nel quale le possibili categorie sono date dalle $H \cdot W$ celle della griglia. Così facendo, la rete ha una prima indicazione riguardo una pluralità di possibili posizioni occupate dall'agente controllato per ogni istante della previsione. Questo permette al modello di individuare una distribuzione di probabilità dei possibili percorsi futuri da parte del soggetto all'interno della griglia 2D, identificando quelli più o meno verosimili.

L'informazione precedente, però, non è sufficiente ad offrire una previsione realmente accurata, perché la scelta di H e W incide sulla risoluzione della griglia, impattando a sua volta sulla qualità delle predizioni. Per migliorare la precisione del sistema, si introduce il Fine Locator (*localizzatore fine*), ovvero un'altra Seq2seq costituita da due ConvRNN (l'encoder e il decoder) aventi uno strato nascosto di dimensioni rispettivamente S_{enc} e S_{dec} e uno stato codificato da una matrice $H \times W \times d$ (come nel CL). A partire dalla sequenza $\boldsymbol{Q}_{1:T_o}$ calcolata dal blocco "to offsets", il FL determina la successione $\hat{\boldsymbol{Q}}_{T_o+1:T_s}$ dei T_p offset futuri per ogni cella della griglia 2D. Contrariamente al localizzatore grossolano, in questo stadio viene risolto un problema di regressione. In altre parole, si ricerca una funzione che approssimi la dipendenza tra la sequenza di offset in ingresso e quella di uscita.
Riassumendo, a valle del CL e del FL sono disponibili due successioni spazio-temporali di lunghezza T_p. Per un istante futuro $t \in [T_o + 1, T_s]$:

- il CL fornisce una heatmap $\hat{\boldsymbol{G}}_t \in \mathbb{R}^{H \times W}$ con le probabilità di passaggio attraverso le celle della griglia 2D al tempo t;
- il FL restituisce $H \cdot W$ vettori 2D che indicano, all'istante t, gli offset del soggetto rispetto ad ogni centro di cella (dato strutturato come $\hat{\boldsymbol{Q}}_t \in \mathbb{R}^{H \times W \times 2}$).

Questi risultati vengono combinati dal blocco "comb" di Figura 19 al fine di determinare le coordinate 2D delle posizioni future più probabili per l'agente controllato al tempo t.
Per una predizione a singolo futuro è possibile ricorrere a:

$$\hat{\boldsymbol{U}}_t = \boldsymbol{C}^{y_t} + \hat{\boldsymbol{Q}}_t^{y_t} \tag{15}$$

dove y_t è l'indice della cella più probabile all'istante t, ricavato dal Coarse Locator, ovvero $y_t = \text{argmax}(\hat{\boldsymbol{G}}_t)$. L'equazione 15, di fatto, descrive una greedy search, nella quale la sequenza completa viene generata concatenando i risultati più probabili di ogni passo temporale.
Per produrre una predizione multi-futuro è quindi necessario ricorrere ad una beam search. In relazione all'esempio fornito nella Sezione 2.4 sull'utilizzo dell'architettura sequence-to-sequence, in questo caso:

- l'insieme delle $H \cdot W$ celle della griglia 2D rappresenta il vocabolario;
- la sequenza delle celle che l'agente controllato attraverserà può essere vista come la frase da generare;

- l'output del Coarse Locator ($\hat{\boldsymbol{G}}_t$) indica la distribuzione di probabilità sul vocabolario di ogni parola (indice di cella) della sequenza (al passo t).

Pertanto, applicando una beam search di parametro F_J su $\hat{\boldsymbol{G}}_{T_o+1:T_s}$, si ricavano:

- una sequenza $Y^{1:F_J}_{T_o+1:T_s} = \{Y^{1:F_J}_{T_o+1}, ..., Y^{1:F_J}_{T_s}\}$ di lunghezza T_p, dove ogni $Y^{1:F_J}_t$ contiene l'indice della cella occupata dalla persona per ciascun futuro predetto;
- un'altra sequenza $\varphi^{1:F_J} = \{\varphi^1, ..., \varphi^{F_J}\}$ di lunghezza F_J, la quale contiene le probabilità stimate per gli scenari futuri determinati.

A questo punto, includendo le informazioni fornite dal Fine Locator, è possibile determinare le F_J traiettorie più probabili per l'agente controllato:

$$\hat{\boldsymbol{U}}^f_t = \boldsymbol{C}^{Y^f_t} + \hat{\boldsymbol{Q}}^{Y^f_t}_t \tag{16}$$

dove l'apice f indica che la grandezza considerata è riferita all'f-esimo futuro. $\hat{\boldsymbol{U}}^{1:F_J}_{T_o+1:T_s}$ e $\varphi^{1:F_J}$ rappresentano gli output finali di Multiverse. L'allenamento dell'intero modello avviene in modo supervisionato: oltre agli ingressi di esempio, vengono forniti i valori di uscita veri sia per il Coarse Locator che per il Fine Locator. Per il classificatore (il CL), in particolare, agli input sequenziali $\boldsymbol{G}_{1:T_o}$ sono associate delle matrici simili (caratterizzate da tutti 0 e un solo 1) $\boldsymbol{G}_{T_o+1:T_s}$. Come funzione di perdita si sfrutta l'entropia incrociata:

$$\mathcal{L}_{CL}(\boldsymbol{G}_{T_o+1:T_s}, \hat{\boldsymbol{G}}_{T_o+1:T_s}) = -\frac{1}{T_p} \sum_{t=T_o+1}^{T_s} \sum_{i=1}^{H} \sum_{j=1}^{W} \boldsymbol{G}^{(i,j)}_t \log\left(\hat{\boldsymbol{G}}^{(i,j)}_t\right). \tag{17}$$

Si procede allo stesso modo con il regressore (il FL), utilizzando come funzione di perdita:

$$\mathcal{L}_{FL}(\boldsymbol{Q}_{T_o+1:T_s}, \hat{\boldsymbol{Q}}_{T_o+1:T_s}) = -\frac{1}{T_p} \sum_{t=T_o+1}^{T_s} \sum_{i=1}^{H} \sum_{j=1}^{W} \text{smooth}\left(a^{(i,j)}_t\right) \tag{18}$$

dove:

$$a^{(i,j)}_t = \frac{|\boldsymbol{Q}^{(i,j)}_{t,x} - \hat{\boldsymbol{Q}}^{(i,j)}_{t,x}| + |\boldsymbol{Q}^{(i,j)}_{t,y} - \hat{\boldsymbol{Q}}^{(i,j)}_{t,y}|}{2} \tag{19}$$

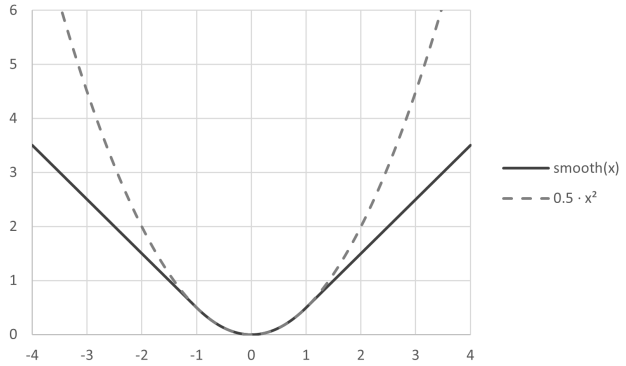

Figura 23: Funzione *smooth*.

mentre la funzione smooth (Figura 23) è una variazione del classico errore quadratico, concepita per essere meno sensibile ad eventuali outliers[1] [10]:

$$\text{smooth}(x) = \begin{cases} 0.5x^2 & \text{se } |x| < 1 \\ |x| - 0.5 & \text{altrimenti.} \end{cases} \tag{20}$$

Le due funzioni di perdita sono combinate linearmente secondo:

$$\mathcal{L} = \mathcal{L}_{CL} + \omega_{\mathcal{L}} \cdot \mathcal{L}_{FL} \tag{21}$$

dove $\omega_{\mathcal{L}} \in \mathbb{R}$ bilancia le funzioni di perdita del Coarse Locator e del Fine Locator.

3.3 TRASFORMAZIONE DELLE TRAIETTORIE

Nella Sezione 3.2 si è discusso il metodo impiegato nel modello Multiverse, e si è evidenziato come questo operi con dati in formato bidimensionale (heatmap e offset 2D). L'obiettivo di questa tesi, però, è lo sviluppo di un algoritmo in grado di lavorare con movimenti umani in 3D, circostanza che impone l'introduzione di uno stadio di conversione dalle traiettorie di giunto (ottenute dalle pose 3D) alle traiettorie 2D compatibili con Multiverse. Le strade percorribili sono due:

1. modificare l'architettura di Multiverse (quindi il Coarse Locator e il Fine Locator) in modo da permettergli di gestire dati tridimensionali in maniera analoga al caso 2D. La griglia 2D viene

1 In un insieme di osservazioni, un *outlier* è un valore che si discosta in maniera sensibile rispetto a tutti gli altri.

portata in 3D e gli offset sono ridefiniti in $\mathbb{R}^3$ anziché in $\mathbb{R}^2$. La convoluzione in 2D è sostituita dalla sua controparte 3D;

2. "mappare" le traiettorie 3D riferite ai giunti in un piano 2D, mantenendo Multiverse nella sua forma originale in due dimensioni. Una possibilità può essere quella di proiettare ogni traiettoria 3D in un piano 2D stabilito a priori (ad esempio, quello individuato dagli assi dei primi due parametri, trascurando così il terzo). Questa soluzione comporta una perdita di informazione rispetto ai dati originali in 3D, fenomeno che può avere conseguenze molto pesanti sia nella fase di addestramento della rete che in quella successiva di funzionamento. Una strategia più evoluta è quella di avvalersi di tecniche di *riduzione della dimensionalità*. Esse permettono, per l'appunto, di ridurre il numero delle dimensioni dello spazio di input (da 3 a 2 in questo caso), garantendo che l'inevitabile perdita di informazione nel processo sia minima.

In questo lavoro si è inizialmente presa la prima via, ovvero quella di ampliare le dimensioni dello spazio percepito dal modello Multiverse, portandolo in 3D. La trasformazione dei dati sulle pose consiste in una semplice scalatura delle traiettorie entro una scena 3D misurata in pixel. Si sono però riscontrate alcune difficoltà piuttosto rilevanti. Innanzitutto, passare da una griglia 2D di dimensioni $H \times W$ ad una 3D $H \times W \times D$ significa incrementare di D volte il numero di celle della griglia. Ne consegue un aumento considerevole dello sforzo computazionale richiesto, soprattutto in corrispondenza del Coarse Locator e del Fine Locator. Un altro problema è dato dal fatto che uno spazio in tre dimensioni è più "profondo" rispetto ad un piano 2D. Per le reti neurali è quindi più difficile apprendere le stesse dipendenze spazio-temporali in maniera corretta. Mantenendo lo stesso dataset di pose umane per allenamento, ciò che appare è che il processo di addestramento del modello in 3D risulta essere estremamente più lento e meno efficace se confrontato con il caso 2D.
Gli ostacoli incontrati hanno successivamente indirizzato la scelta verso la conversione delle traiettorie di giunto dallo spazio tridimensionale ad uno bidimensionale. Il metodo applicato è esposto nel seguito.
Nel passaggio da 3D a 2D vengono effettuate due trasformazioni:

- viene dapprima applicata una tecnica di riduzione della dimensionalità al fine di ridurre da 3 a 2 il numero di parametri necessario ad esprimere una posizione o una rotazione nello spazio;
- in seguito, le traiettorie 2D ottenute al punto precedente subiscono una scalatura, in modo da farle rientrare nella scena video $V_H \times V_W$ (in pixel) interpretabile da Multiverse.

In generale, gli algoritmi di riduzione della dimensionalità permettono di diminuire il numero di variabili necessarie a descrivere un set di dati, minimizzando la conseguente perdita di informazione. Sono ampiamente utilizzati in tutti quei campi in cui è richiesta l'elaborazione di dati ad elevata dimensionalità (ad esempio, tracce audio o foto digitali). Esistono molti metodi adatti allo scopo, quello impiegato in questo lavoro prende il nome di *analisi delle componenti principali* (*principal component analysis*, PCA) [32]. La PCA sfrutta una trasformazione lineare per proiettare i dati in uno spazio di dimensione minore (Figura 24).

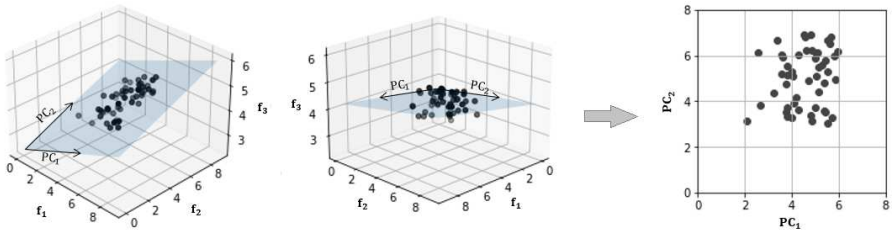

Figura 24: Analisi delle componenti principali: rappresentazione grafica.

Sia $\boldsymbol{X}_{1:B} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, ..., \boldsymbol{X}_B\}$ il dataset per l'allenamento, costituito da una serie di B campioni $\boldsymbol{X}_h \in \mathbb{R}^{n_V}$ che sono funzione di n_V variabili (in questo caso 3). L'obiettivo è ricondurre $\boldsymbol{X}_{1:B}$ ad una rappresentazione $\boldsymbol{U}_{1:B}$ che utilizzi un numero di componenti n_C minore (in questo caso 2). Per fare questo, si determina la matrice delle covarianze $\boldsymbol{\Sigma} \in \mathbb{R}^{n_V \times n_V}$, il cui elemento in posizione (i, j) è la covarianza tra la i-esima e la j-esima variabile del dataset originale:

$$\boldsymbol{\Sigma}_{i,j} = \frac{1}{B} \sum_{h=1}^{B} (\boldsymbol{X}_{h,i} - \mu_i)(\boldsymbol{X}_{h,j} - \mu_j) \tag{22}$$

dove $\mu_i \in \mathbb{R}$ è la media della i-esima variabile. Siano $\lambda_1, \lambda_2, ..., \lambda_{n_V}$ gli autovalori di $\boldsymbol{\Sigma}$, tutti positivi per la natura della matrice stessa. L'autovalore λ_i, in particolare, esprime "il contributo sulla varianza complessiva" della i-esima variabile rispetto alle altre. L'idea della PCA

è di utilizzare solamente i primi n_C autovalori (in ordine decrescente) per ricostruire i dati originali, in modo da ridurre la dimensionalità massimizzando la varianza e riducendo così al minimo la perdita di informazione. Gli autovettori associati a questo sottoinsieme di autovalori rappresentano le cosidette *componenti principali*, ovvero gli assi del nuovo spazio a dimensione ridotta. Concatenando in riga le componenti principali si ottiene la matrice delle componenti principali ($\mathbf{\Gamma} \in \mathbb{R}^{n_C \times n_V}$), dipendente unicamente dai dati.
La trasformazione introdotta dalla PCA per la riduzione è dunque:

$$\mathbf{\Psi}_h = (\boldsymbol{X}_h - \boldsymbol{\mu}) \cdot \mathbf{\Gamma}^{\mathrm{T}} \tag{23}$$

dove $\mathbf{\Psi}_h \in \mathbb{R}^{n_C}$ e $\boldsymbol{\mu} \in \mathbb{R}^{n_V}$ è il vettore che include le medie μ_i per ogni variabile i. Attraverso l'*explained variance ratio* (EVR), si può quantificare la perdita di informazione indotta dalla riduzione:

$$\mathrm{EVR}_i = \frac{\lambda_i}{\sum_{j=0}^{n_V} \lambda_j} \tag{24}$$

il quale indica il contributo (in termini relativi, quindi da 0% a 100%) della i-esima variabile dello spazio orginale sulla totale varianza del dataset. Se l'EVR totale calcolato sulle prime n_C componenti è alto (vicino a 100%), significa che l'applicazione della PCA implica una piccola perdita di informazione.
Una volta ridotta la dimensione delle traiettorie, occorre scalarle entro lo spazio dei video pixel, come richiesto da Multiverse. La relazione che permette di passare dal dataset ridotto ($\mathbf{\Psi}_{1:B}$) a quello scalato ($\boldsymbol{U}_{1:B}$) è:

$$\boldsymbol{U}_h = (\mathbf{\Psi}_h - \boldsymbol{L}) \cdot \boldsymbol{K} \tag{25}$$

in cui $\boldsymbol{K} \in \mathbb{R}^{2\times 2}$ è una matrice diagonale avente come valori gli "zoom" da applicare ai due assi dello spazio ridotto (che nel seguito identificano altezza e larghezza dei video frames), mentre $\boldsymbol{L} \in \mathbb{R}^2$ rappresenta una traslazione che renda positivi tutti i valori del dataset $\boldsymbol{U}_{1:B}$. $\boldsymbol{K}$ e $\boldsymbol{L}$ dipendono dal range di variazione dei dati, ovvero dai minimi e dai massimi, oltre che dalla dimensione $V_H \times V_W$ della scena video. Ovviamente, a differenza della PCA, l'operazione di scalatura non causa perdita di informazioni.
Il risultato finale del procedimento di trasformazione da 3D a 2D è una traiettoria compatibile con Multiverse, il quale si occupa di generarne un set di andamenti futuri ($\hat{\boldsymbol{U}}^{1:F_J}_{T_o+1:T_s}$). Ciascuno di essi va in seguito

riconvertito in 3D, applicando la relazione di trasformazione inversa ricavabile dalle Equazioni (23) e (25):

$$\hat{\boldsymbol{X}}_h^f = \left(\boldsymbol{U}_h \boldsymbol{K}^{-1} + \boldsymbol{L}\right) \boldsymbol{\Gamma} + \boldsymbol{\mu}. \tag{26}$$

A questo punto, si ha a disposizione un metodo per effettuare la predizione su traiettorie 3D.

3.4 APPLICAZIONE DEI VINCOLI ANATOMICI

Una volta compreso come le traiettorie 3D future vengono generate, occorre trovare un procedimento per verificare l'eventuale presenza di valori non compresi entro i range consentiti. Il corpo umano, come evidenziato in Sezione 1.3, è infatti soggetto a dei vincoli anatomici che limitano gli intervalli di rotazione dei giunti. Si fa notare che la suddivisione dei movimenti umani in traiettorie di giunto indipendenti tra loro rende indifferente l'inserimento di questa fase a monte o a valle del blocco di combinazione di Figura 17.
La procedura descritta nel seguito trae ispirazione dall'algoritmo di cinematica inversa (IK, *Inverse Kinematics*) utilizzato in OpenSim [6]. Quest'ultimo è un software open source impiegato nel campo della biomeccanica per la modellazione di sistemi muscolo-scheletrici e per la simulazione dinamica del movimento. Il metodo IK di OpenSim fornisce una stima anatomicamente corretta del movimento umano a partire dai dati ottenuti da un generico sistema di campionamento dello stesso (*motion capture system*).
Per ogni giunto dello scheletro umano, i vincoli anatomici considerati in questo lavoro sono di tipo statico ed espressi come rotazioni. Si rende necessario in primo luogo convertire le rotazioni $\hat{\boldsymbol{X}}_t$ predette in precedenza (cioè i punti delle traiettorie di giunto) dalla rappresentazione asse-angolo alle terne di Eulero ($\hat{\boldsymbol{\alpha}}_t = \{\hat{\alpha}_{t,1}, \hat{\alpha}_{t,2}, \hat{\alpha}_{t,3}\}$), in modo da gestire separatamente le singole rotazioni elementari attorno agli assi X, Y e Z. I vincoli si esprimono nel modo seguente:

$$\begin{cases} L_1 \leq \hat{\alpha}_1 \leq H_1 \\ L_2 \leq \hat{\alpha}_2 \leq H_2 \\ L_3 \leq \hat{\alpha}_3 \leq H_3 \end{cases} \tag{27}$$

dove i limiti L_i e H_i sono caratteristici di ogni giunto. Ogni angolo $\hat{\alpha}_i$ rilevato oltre i limiti viene riportato entro l'intervallo concesso:

$$\tilde{\alpha}_{t,i} = \begin{cases} L_i & \text{se } \hat{\alpha}_{t,i} < L_i \\ H_i & \text{se } \hat{\alpha}_{t,i} > H_i \\ \hat{\alpha}_{t,i} & \text{altrimenti.} \end{cases} \tag{28}$$

Infine, il vettore $\tilde{\boldsymbol{\alpha}}_t = \{\tilde{\alpha}_1, \tilde{\alpha}_2, \tilde{\alpha}_3\}$ è riconvertito in formato asse-angolo, ottenendo il punto vincolato $\tilde{\boldsymbol{X}}_t$. Applicando lo stesso procedimento per tutti i punti, si ottengono delle traiettorie vincolate, ovvero l'output di questo stadio.

Non è facile assegnare dei valori univoci a L_i e H_i che siano plausibili per tutto il genere umano, e SMPL non fornisce alcuna indicazione su di essi. La via più semplice è quella di rilevare, per ogni parametro delle rotazioni delle pose AMASS (convertite nella notazione di Eulero), i massimi e i minimi raggiunti nel dataset di allenamento.

Il problema da risolvere è la rilevazione della miglior sequenza di Eulero per la rappresentazione delle rotazioni di ogni giunto. In questo lavoro si sono seguite le raccomandazioni fornite dall'International Society of Biomechanics (ISB) [46, 47, 48], la quale consiglia di adottare la sequenza ZXY con riferimento alla definizione degli assi di Figura 25.

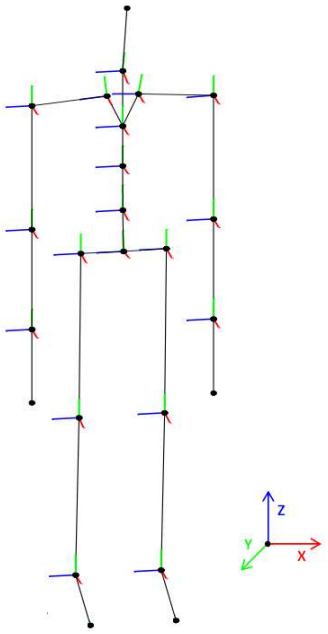

Figura 25: Sistemi di riferimento XYZ adottati dall'ISB [50].

L'ultima fase dell'algoritmo proposto consiste nella combinazione delle traiettorie di giunto predette (e vincolate), allo scopo di acquisire un certo numero (F_B) di sequenze future di movimento umano.
Il metodo descritto finora ha permesso di ricavare $J+2$ set contenenti ognuno F_J traiettorie di giunto future ($\hat{\boldsymbol{X}}^{1:F_J}_{0,T_o+1:T_s}$, $\hat{\boldsymbol{X}}^{1:F_J}_{1,T_o+1:T_s}$, ..., $\hat{\boldsymbol{X}}^{1:F_J}_{20,T_o+1:T_s}$, $\hat{\boldsymbol{X}}^{1:F_J}_{21,T_o+1:T_s}$, $\hat{\boldsymbol{X}}^{1:F_J}_{T,T_o+1:T_s}$), ciascuna associata una certa probabilità ($\varphi_0^{1:F_J}$, $\varphi_1^{1:F_J}$, ..., $\varphi_T^{1:F_J}$). Imponendo $F_J = 1$, si avrebbero esattamente $J+2$ traiettorie $\hat{\boldsymbol{X}}_{j,T_o+1:T_s}$ definite in $\mathbb{R}^3$, una per ogni giunto più due per orientazione e traslazione del corpo. In tale circostanza, la combinazione si faciliterebbe molto: il movimento umano di output sarebbe la semplice concatenazione delle $J+2$ successioni temporali $\hat{\boldsymbol{X}}_{j,T_o+1:T_s}$, cioè $\hat{\boldsymbol{P}}_{T_o+1:T_s} = \left[\hat{\boldsymbol{X}}_{0,T_o+1:T_s}, \hat{\boldsymbol{X}}_{1,T_o+1:T_s}, ..., \hat{\boldsymbol{X}}_{T,T_o+1:T_s}\right]$, ed al risultato ottenuto sarebbe associata la probabilità $\Phi = \varphi_0 \cdot \varphi_1 \cdot ... \cdot \varphi_T$.
Con $F_J > 1$ il problema si complica, perché per ogni giunto sono disponibili F_J sequenze di moto. Potenzialmente, il numero di scenari di movimento umano generabili con questi dati è pari a $(J+2)^{F_J}$ (se $J = 18$ e $F_J = 20$, il totale è $20^{20} \approx 10^{26}$). Appare evidente che non ha molto senso verificare ogni singola opzione possibile, rendendo pertanto preferibile ricorrere ad un metodo alternativo. Si applica nuovamente una beam search, questa volta sulle traiettorie $\hat{\boldsymbol{X}}_{j,T_o+1:T_s}$. Il movimento umano, interpretabile come un set di traiettorie, rappresenta in questo caso la sequenza di output da generare, avente lunghezza nota ($J+2$). Il j-esimo elemento di questa successione è selezionabile dal set dei F_J futuri relativi al giunto di indice j, in base alle probabilità stimate. Il numero di scenari futuri da calcolare è uguale al parametro F_B di questa beam search; per una predizione a singolo futuro basta porre $F_B = 1$, impegando così una greedy search.
A differenza della beam search applicata all'interno di Multiverse per ricavare una predizione multi-futuro, nelle sequenze qui intese gli elementi in posizione j non dipendono da quelli in posizione $j-1$, per via dell'ipotesi fatta sull'indipendenza delle traiettorie dei giunti. Nonostante ciò, questo algoritmo di ricerca si rivela utile e adeguato all'applicazione. Al termine del procedimento, si hanno a disposizione F_B set di $J+2$ traiettorie, ognuno associato ad una probabilità. L'output è fornito tramite due sequenze:

- $\hat{\boldsymbol{P}}^{1:F_B}_{T_o+1:T_s}$, le vere e proprie pose multi-futuro;

- $\Phi^{1:F_B} = \{\Phi^1, \Phi^2, ..., \Phi^{F_B}\}$, che include la probabilità stimata per ciascuno dei futuri predetti.

Questo è il risultato finale del metodo di anticipazione presentato in questa tesi.

4

SPERIMENTAZIONE

In questo capitolo vengono valutate le prestazioni del sistema di anticipazione studiato. Dopo una breve spiegazione di come i dati AMASS sono stati maneggiati, si forniscono i risultati ottenuti nella predizione di movimenti umani.

4.1 SETUP E PREPROCESSING DEI DATI

Per l'allenamento e la successiva verifica del modello di predizione sviluppato si sfrutta il dataset AMASS introdotto nella Sezione 1.4. L'enorme mole di dati contenuta in quest'ultimo, però, rende sensibilmente lunghi i tempi di addestramento. Pertanto, solo una piccola porzione di essi viene effettivamente impiegata per ricavare i risultati esposti nel seguito. In particolare, si sono selezionati casualmente 817 motion files (circa il 7.3% di AMASS) per l'allenamento del modello ed altri 293 per collaudarlo (il 2.6% del dataset). Inoltre, ogni campione di movimento, riferito inizialmente a 60 Hz, viene portato a 5 Hz (scartando 11 pose frames ogni 12). Queste due procedure consentono di ridurre i tempi richiesti per svolgere l'allenamento.
I dati per l'addestramento vengono raccolti da ogni motion file ridotto a 5 Hz facendo scorrere una finestra di $T_o + T_p$ frame sulle sequenze di movimento. Se la durata del campione è T, allora vengono prodotte $T + 1 - T_o - T_p$ traiettorie 3D per ciascun giunto (più una per la traslazione e una per l'orientazione globale) di lunghezza $T_o + T_p$. Queste sono trasformate ed in seguito fornite alla rete per l'allenamento supervisionato (come descritto in Sezione 3.2).
Il test del sistema, invece, avviene fornendogli soltanto le prime T_o pose di un motion file che ne contiene T. Si valuta, in particolare, la capacità del modello di generare i movimenti futuri corretti a partire dalle sole pose iniziali.

Per la configurazione del modello, si sono perlopiù mantenute le impostazioni consigliate dagli sviluppatori di Multiverse.
Per quanto riguarda l'architettura delle reti interne a quest'ultimo, si sono impostate a $S_{enc} = 256$ e $S_{dec} = 256$ le dimensioni degli strati nascosti del Coarse e del Fine Locator, con una dimensione degli stati latenti di $d = 32$ (per singola cella 2D). Il kernel della convoluzione 2D è una matrice 3×3. Come funzione di attivazione delle celle nelle ConvRNN, si utilizza la tangente iperbolica. Il problema di minimizzazione dell'addestramento viene risolto con l'ottimizzatore AdaDelta [51], con un learning rate iniziale di $\eta_0 = 0.3$ ed un suo decadimento pari a $\rho = 0.95$. Il coefficiente di bilanciamento delle funzioni di perdita di classificatore e regressore è posto a $\omega_{\mathcal{L}} = 0.2$.
I fotogrammi elaborati da Multiverse hanno una risoluzione di $V_H \times V_W$, con $V_H = 1080$ px e $V_W = 1920$ px. Per la griglia 2D, si è mantenuta la dimensione consigliata di $H \times W$, con $H = 18$ celle e $W = 32$ celle. Ogni cella della griglia copre quindi un'area di 60×60 pixel.
Il modello completo viene allenato in modalità supervisionata fornendogli degli esempi di coppie input-output, dove gli ingressi e le uscite sono campioni di movimento di lunghezza rispettivamente $T_o = 8$ e $T_p = 12$ (espresse in numero di pose frames), corrisponenti a durate di 1.6 e 2.4 secondi (con un campionamento a 5 Hz).
In riferimento al sub-dataset utilizzato, lo spazio in memoria richiesto per contenere queste informazioni (utilizzando una codifica a virgola mobile *float32* per ogni parametro) si aggira intorno ai 12 GB per ogni terna (traslazione o rotazione) del corpo umano. Con $J + 2 = 20$ modelli da allenare, il totale è di circa 240 GB. Il numero di epoche selezionato è pari a 20. Disponendo di una GPU NVIDIA GeForce RTX 2080 e imponendo un batch size di 25 esempi/passo (che si è visto essere il massimo sostenibile dal calcolatore), il processo di addestramento di un singolo modello Multiverse (relativo ad un giunto) impiega circa 20 ore. Con $J + 2 = 20$ modelli da allenare, il tempo totale necessario è di 16.7 giorni.
La sequenza di uscita, potenzialmente, non ha limiti di estensione. Tuttavia, si è preferito contenerla entro una lunghezza massima di 48 frame (9.6 secondi), perché per più ampi orizzonti le stime sono poco attendibili.
Il numero di futuri scelto per le traiettorie di giunto è di $F_J = 20$,

mentre per gli output finali di movimento completo si pone $F_B = 30$.

4.3 EFFETTO DELLA RIDUZIONE 3D/2D

Come prima analisi, si valuta la perdita di informazione indotta dalla riduzione delle pose 3D in uno spazio bidimensionale. L'algoritmo di conversione è quello esposto nella sezione 3.3.
La trasformazione da uno spazio 3D a uno 2D avviene tramite il mantenimento di 2 componenti principali rispetto alle 3 variabili iniziali, alle spese della componente meno significativa che viene trascurata. Di conseguenza, si può quantificare la perdita di informazione calcolando l'explained variance ratio correlato alla variabile ignorata. In Tabella 2 sono riportati gli EVR calcolati per tutte e tre le variabili di ogni rotazione e traslazione, mentre in Figura 26 è tracciato il grafico a barre relativo all'EVR della terza componente (EVR_3, quella trascurata). Le trasformazioni migliori, cioè con perdite di piccola entità, sono quelle associate ad un basso EVR_3.

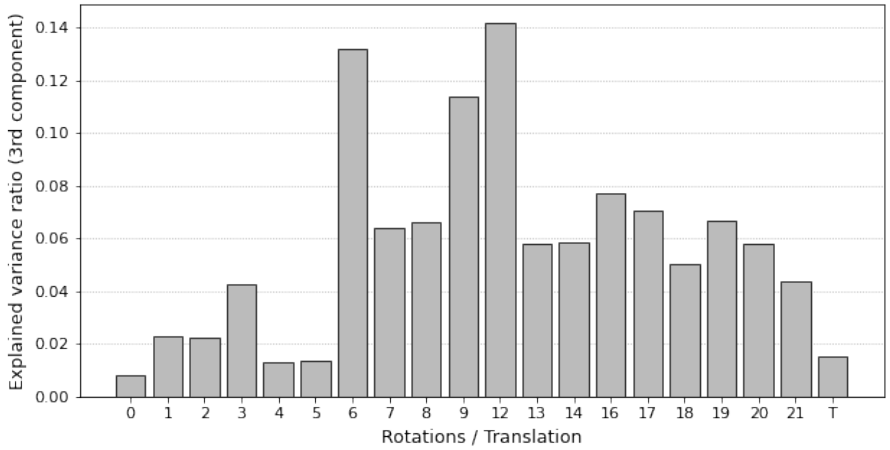

Figura 26: Perdita di informazione indotta dalla PCA (EVR_3).

Com'era preventivabile, le minori perdite di dati si hanno in corrispondenza delle rotazioni dei giunti che hanno meno gradi di libertà. Le rotazioni delle ginocchia (indici 4 e 5), che sono giunti ad un solo grado di libertà, sono proiettabili nello spazio ridotto con un'esigua perdita di informazione. I giunti più sfavoriti nella trasformazione risultano essere la colonna vertebrale (indici 6 e 9) e il collo (indice 12), tutti a tre gradi di libertà. Le anche (indici 1 e 2) paiono sfuggire da questa norma, ma la spiegazione è data dal fatto che questi giunti

a tre GdL sono in genere "poco varianti" durante il movimento: le rotazioni tipiche sono la flessione e l'adduzione, mentre la torsione si verifica in maniera più sporadica. Considerazioni simili possono essere fatte anche sulle spalle (indici 16 e 17).

Indice giunto	Nome giunto	Num. gradi di libertà	EVR_1 (%)	EVR_2 (%)	**EVR_3 (%)**
0	*Orientazione*	3	73.6	25.6	**0.8**
1	anca sx	3	85.7	12.0	**2.3**
2	anca dx	3	87.8	10.0	**2.2**
3	col. vert. 1	3	83.1	12.7	**4.2**
4	ginocchio sx	1	87.7	11.0	**1.3**
5	ginocchio dx	1	90.9	7.7	**1.4**
6	col. vert. 2	3	67.4	19.5	**13.2**
7	caviglia sx	2	76.3	17.3	**6.4**
8	caviglia dx	2	75.7	17.7	**6.6**
9	col. vert. 3	3	66.2	22.4	**11.4**
12	collo	3	53.2	32.6	**14.2**
13	clavicola sx	2	60.4	33.8	**5.8**
14	clavicola dx	2	57.8	36.4	**5.8**
16	spalla sx	3	67.7	24.6	**7.7**
17	spalla dx	3	73.5	19.5	**7.1**
18	gomito sx	2	71.7	23.3	**5.0**
19	gomito dx	2	70.8	22.6	**6.7**
20	polso sx	2	59.3	34.9	**5.8**
21	polso dx	2	61.1	34.6	**4.4**
T	*Traslazione*	3	83.6	14.8	**1.5**

Tabella 2: Explained variace ratios nella trasformazione 3D/2D.

L'orientazione e la traslazione globale del corpo, invece, costituiscono delle eccezioni rispetto a quanto scritto in precedenza. Sebbene queste due trasformazioni siano a 3 gradi di libertà e largamente variabili (non hanno alcun vincolo), la perdita di informazione causata dalla loro riduzione è molto più bassa rispetto a qualsiasi altro giunto. La traslazione, in particolare, sembra essere ben descrivibile anche

con un solo parametro (per via dell'alto rapporto EVR_1/EVR_2). La spiegazione è duplice. Innanzitutto, i parametri associati risentono del fenomeno descritto in precedenza. La traslazione, ad esempio, è per la verità rappresentabile con buona approssimazione utilizzando solamente 2 assi orizzontali, ignorando l'informazione sulla verticalità (la quale si mantiene abbastanza costante durante tutto il moto). Un discorso simile è valido anche per l'orientazione: i soggetti rimangono tipicamente in piedi e rivolti verso la stessa direzione. Rispetto alle rotazioni dei giunti, però, c'è un ulteriore aspetto da considerare, dovuto alla natura di AMASS. Il dataset presenta un'ampia gamma di movimentazioni umane in 3D, ma la sezione numericamente più rilevante è composta da movimenti "sul posto" compiuti con una postura eretta. In questi campioni, le coordinate relative alla posizione e all'orientazione del corpo rimangono approssimativamente costanti durante tutta la durata del moto. I motion files che descrivono movimenti più variegati sono una minoranza, pertanto hanno una ridotta influenza sulla totale varianza di AMASS. Con queste condizioni, un EVR_3 basso non fornisce certezze sulla qualità della riduzione (intesa in termini di basse perdite), bensì si rivela più affidabile nella valutazione dell'effettiva differenziazione del dataset. Una selezione mirata delle sequenze da utilizzare potrebbe fornire risultati più accurati. Va detto, comunque, che i parametri descrittivi della posizione e dell'orientazione globale sono fortemente dipendenti dall'ambiente circostante, fattore trascurato in questo lavoro.
Gli aspetti descritti in precedenza consentono di affermare che:

- per quanto riguarda le rotazioni dei giunti, la riduzione non ha in generale un effetto troppo pesante, anche se l'eccessivo errore indotto nei giunti della colonna vertebrale potrebbe comportare qualche difficoltà in più nella predizione delle pose corrette dall'addome in su;

- le trasformazioni 3D/2D delle traiettorie di traslazione e orientazione globale non alterano particolarmente il contenuto informativo di AMASS, ma questo fatto è correlato più che altro al tipo di dati contenuti in esso. È possibile che il dataset selezionato sia poco rappresentativo degli spostamenti di una persona all'interno della scena 3D, concentrandosi prevalentemente sulle posture assunte dall'individuo. Trascurando questo fatto, può accadere che le predizioni di movimenti ampi e complessi risultino essere

meno accurate. Ad ogni modo, la stima delle pose future non ne risente, dato che vengono considerate le rotazioni relative dei giunti. Una possibile soluzione potrebbe essere quella di modificare manualmente il dataset, assegnando al root delle traiettorie più varie, generabili manualmente o derivabili da altri dataset. In entrambi i casi, quest'attività deve essere svolta tenendo conto del contesto in cui andrebbe ad operare il sistema.

Nel complesso, seppur con qualche piccola criticità, si può comunque affermare che l'analisi delle componenti principali è adeguata a svolgere la trasformazione dei dati richiesta. A scopo informativo, la quantità totale di informazione perduta con l'applicazione della PCA (ricavabile come media aritmetica di tutti gli EVR_3) è pari al 5.7%.

4.4 RISULTATI IN 2D

Sono esposti nel seguito i risultati ottenuti per mezzo del modello Multiverse nella predizione delle traiettorie 2D relative ad ogni rotazione e traslazione del corpo umano.
Per prima cosa, si vuole valutare la varietà degli scenari futuri predetti, generati a partire da una singola traiettoria osservata di durata T_o. In una predizione multi-futuro sono presenti F_J traiettorie future possibili di lunghezza T_p. È quindi possibile determinare una distanza euclidea per ognuna delle $F_J(F_J-1)/2$ coppie di traiettorie, per poi calcolare la media di tali valori:

$$\text{DI}_t = \frac{2}{F_J(F_J-1)} \sum_{f_1=1}^{F_J-1} \sum_{f_2=f_1+1}^{F_J} \|\boldsymbol{u}_{T_o+t}^{f_1} - \boldsymbol{u}_{T_o+t}^{f_2}\|. \tag{29}$$

DI_t rappresenta un "indice di diversità" della predizione multi-futuro riferito al t-esimo passo, e si misura in pixel (px). Si può infine calcolare una media dei DI_t per t che va da T_o+1 a T:

$$\text{ADI}_T = \frac{1}{T} \sum_{t=1}^{T} \text{DI}_t. \tag{30}$$

Nelle Figure 27 e 28 sono esposti rispettivamente i DI_t per $t \in [1, 25]$ ed i ADI_T per $T = 6$ e $T = 12$.
Si nota che le traiettorie multi-futuro non sono per nulla variegate. Con l'unica esclusione del primo passo di predizione, in cui i percorsi effettivamente si biforcano (DI_1 medio di 19 px), i futuri generati da

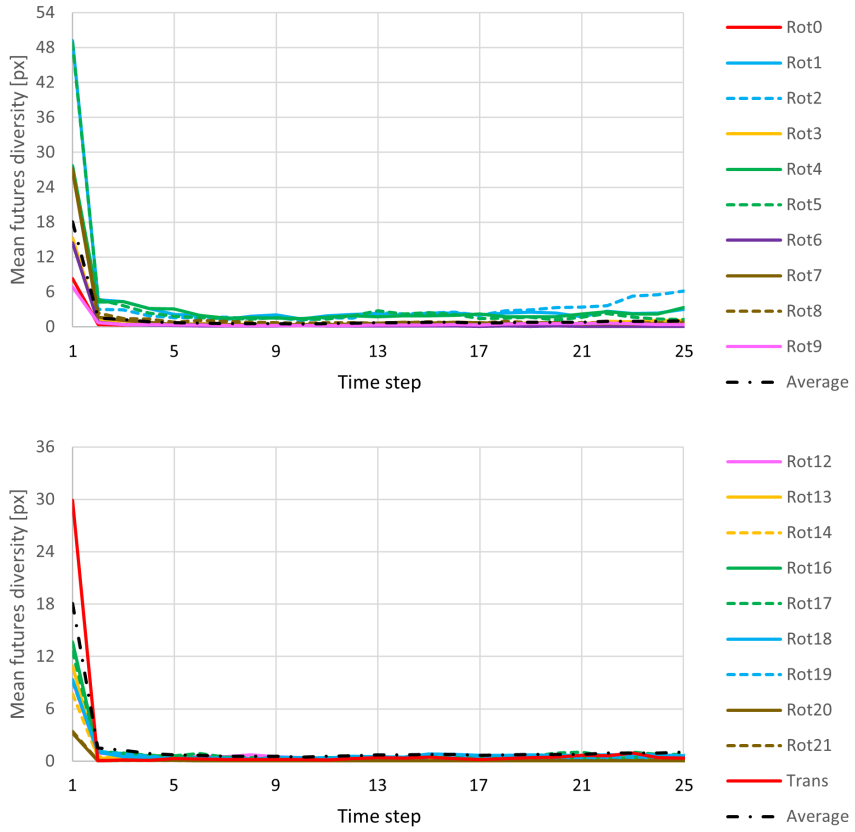

Figura 27: Indici di diversità delle predizioni multi-futuro per ogni giunto (andamento temporale).

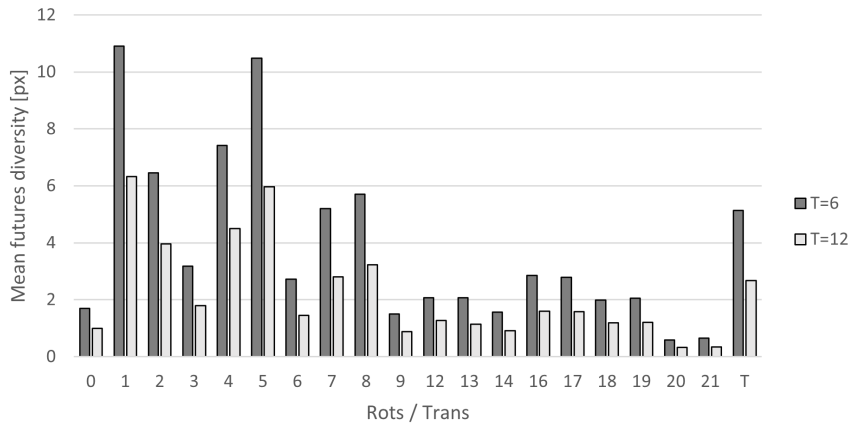

Figura 28: Indici di diversità delle predizioni multi-futuro per ogni giunto (media su 6 e 12 passi di predizione).

Multiverse risultano essere molto simili tra loro dal secondo passo di predizione in poi, con un ADI_6 medio di 3.9 px (si ricorda che il piano dei fotogrammi ha una dimensione di 1080×1920 pixel). I giunti che restituiscono una gamma maggiore (ma comunque piccola) di futuri distinti sono quelli degli arti inferiori: anche, ginocchia e caviglie ($\text{DI}_1 > 20$ px, $\text{ADI}_6 > 6$ px). In altre parole, i modelli Multiverse fanno fatica ad "immaginare" dei percorsi alternativi rispetto a quello che ritengono più probabile. La causa di un così ristretto assortimento di scenari futuri va ricercata nel Coarse Locator e nella sua attività di generazione di heatmap. Nella maggior parte dei casi, infatti, esse risultano essere "degeneri", con un accumulo di probabilità (oltre il 97%) sulla stessa cella e una leggera dispersione su quelle attorno. Gli output del Coarse Locator tendono quindi ad assomigliare troppo agli input, cioè a matrici di tutti 0 e un solo 1, che sono state utilizzate anche nel processo di allenamento supervisionato. Una possibile causa del fenomeno può essere ricondotta ad AMASS. Per una data sequenza iniziale di pose, potrebbe non essere presente una così vasta scelta di movimenti futuri distinti, rendendo il dataset poco adatto alla predizione multi-futuro. Per alcuni giunti, che per loro natura non hanno grandi variazioni durante il movimento umano, questo fatto è da considerarsi normale. Le anomalie maggiori sono legate alla traslazione e all'orientazione del corpo, che dovrebbero invece essere più varianti. Ancora una volta, una modifica manuale dei dati AMASS potrebbe migliorare i risultati anche sotto questo punto di vista.
Nel seguito, si valutano gli errori quantitativi di predizione in 2D. In prima istanza, si può definire per ogni istante di predizione t un *displacement error* (DE_t), inteso come la distanza euclidea tra il t-esimo punto della traiettoria vera ($\boldsymbol{u}_t$) e il t-esimo punto della traiettoria predetta più simile a quella reale ($\hat{\boldsymbol{u}}_t^{\mathcal{F}}$, dove $\mathcal{F}$ designa l'indice del "miglior futuro predetto"):

$$\text{DE}_t = \|\boldsymbol{u}_{T_o+t} - \hat{\boldsymbol{u}}_{T_o+t}^{\mathcal{F}}\|. \tag{31}$$

A partire dal DE, si calcola un *average displacement error* riferito ad un orizzonte di previsione di durata T:

$$\text{ADE}_T = \frac{1}{T}\sum_{t=1}^{T} \text{DE}_t. \tag{32}$$

DE e ADE sono definiti come distanze in un fotogramma, quindi sono espressi in pixel. In Figura 29 sono riportati gli andamenti temporali dei DE_t, mentre in Figura 30 si presentano gli ADE_6 e gli ADE_{12}.

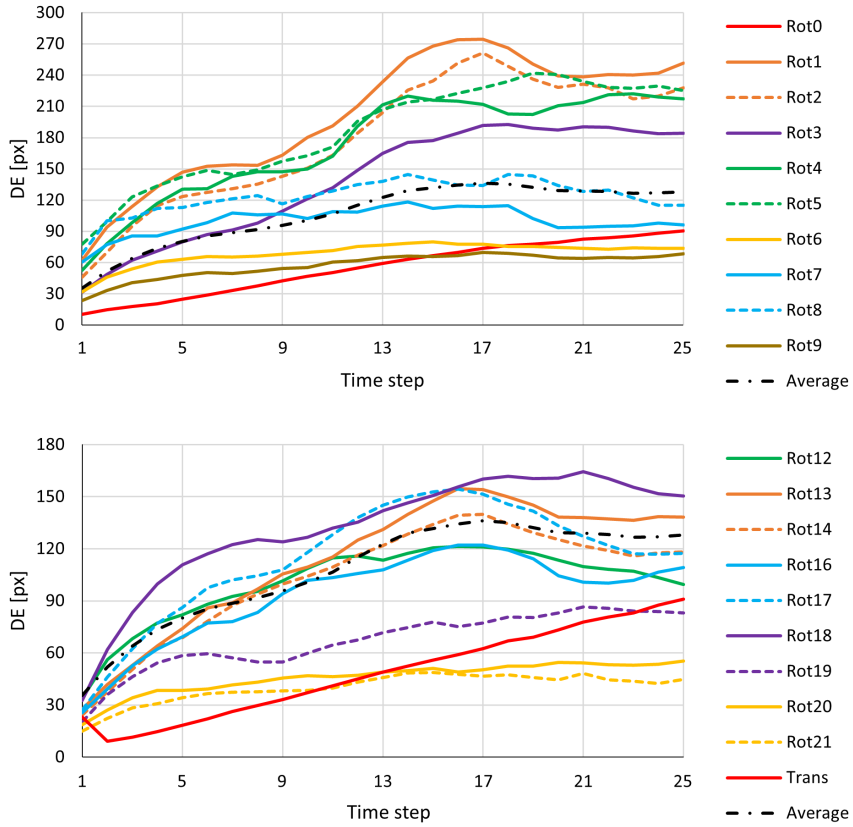

Figura 29: Displacement errors per ogni rotazione e traslazione.

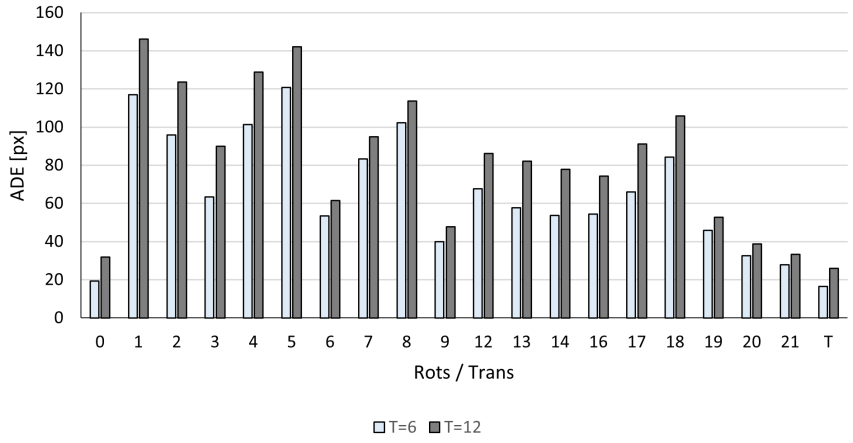

Figura 30: Average displacement errors per ogni rotazione e traslazione (riferito a 6 e a 12 passi di predizione).

Come si può notare, l'errore medio per ogni giunto tende ad aumentare con il tempo fino al passo $t \approx 15$, oltre il quale si stabilizza. Una piccola anomalia si riscontra nella predizione della traiettoria 2D relativa alla traslazione del corpo, caratterizzata da un picco di errore al primo passo di predizione (curva rossa in Figura 29). Considerate le dimensioni del piano immagine sul quale sono proiettate le traiettorie 2D (1080×1920 px), è possibile affermare che gli errori di stima ottenuti sono accettabili. In particolare, le prestazioni peggiori si hanno in corrispondenza dei giunti degli arti inferiori.

4.5 RISULTATI IN 3D

Una volta generate le traiettorie multi-futuro in due dimensioni, si applica la trasformazione inversa illustrata nella Sezione 3.3 al fine di ottenere delle traiettorie tridimensionali espresse nello spazio metrico (per la traslazione del corpo) o in quello della notazione asse-angolo (per l'orientazione del corpo e le rotazioni dei giunti). Rispetto ai risultati in 2D esposti nella Sezione 4.4, qui entra in gioco anche l'effetto della perdita di informazione indotta dalla PCA.
La metrica più semplice per la quantificazione degli errori è il displacement error in 3D:

$$\mathrm{DE}_t = \|\boldsymbol{X}_{T_o+t} - \hat{\boldsymbol{X}}^{\mathcal{F}}_{T_o+t}\| \tag{33}$$

($\mathcal{F} \in [1, F_J]$ è l'indice della miglior traiettoria nel set multi-futuro) il quale, mediato su un orizzonte di predizione lungo T, resituisce l'average displacement error in 3D:

$$\mathrm{ADE}_T = \frac{1}{T}\sum_{t=1}^{T} \mathrm{DE}_t. \tag{34}$$

Essendo DE e ADE delle distanze euclidee, essi si misurano in metri nel caso della traslazione e in radianti nel caso delle rotazioni (perché la norma di un vettore in notazione asse-angolo è uguale alla misura della rotazione associata). Gli andamenti di DE sono riportati in Figura 31 (traslazione, in centimetri) e in Figura 32 (rotazioni, in gradi). Nelle Figure 33 e 34 sono invece esposti gli ADE riferiti a 6, 12, 18 e 24 passi di predizione.
Si analizza in primo luogo l'errore di posizionamento del corpo. Anche in tre dimensioni è visibile il fenomeno riscontrato in 2D, in cui

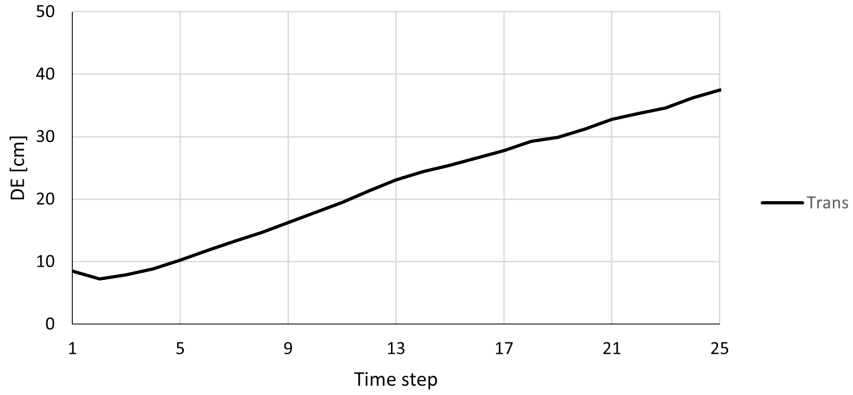

Figura 31: Displacement error 3D nella predizione della traslazione del corpo.

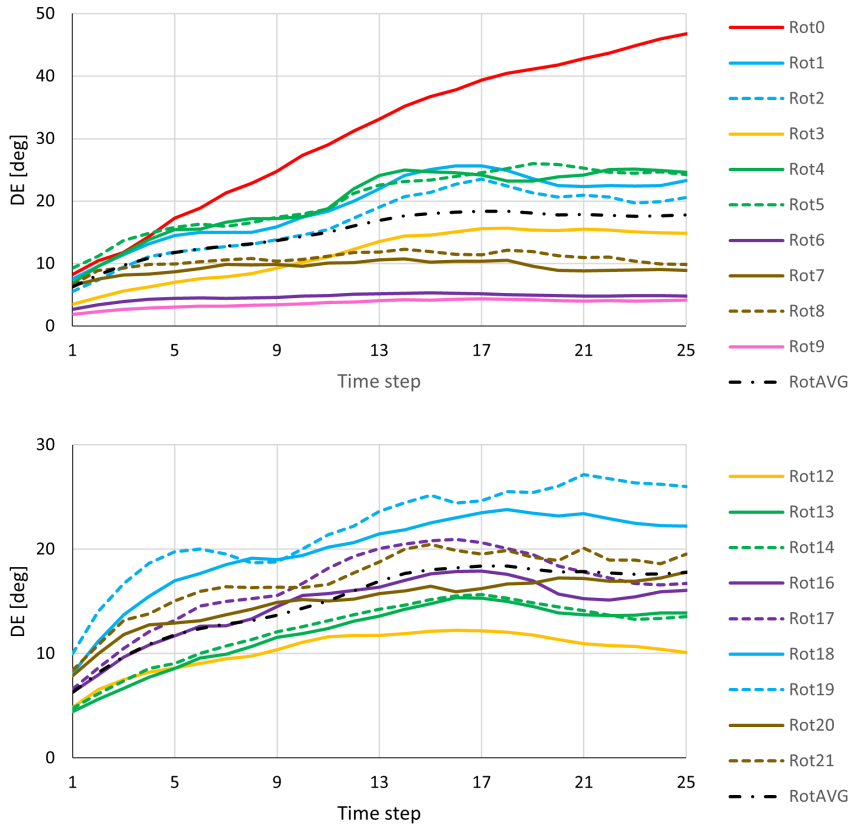

Figura 32: Displacement error nella predizione delle rotazioni 3D.

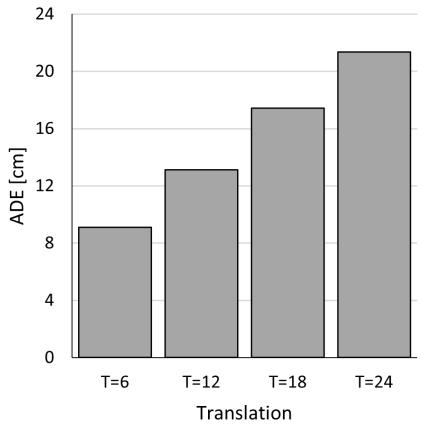

Figura 33: Errore euclideo medio nella predizione della traslazione 3D in un orizzonte di 6, 12, 18 e 24 passi.

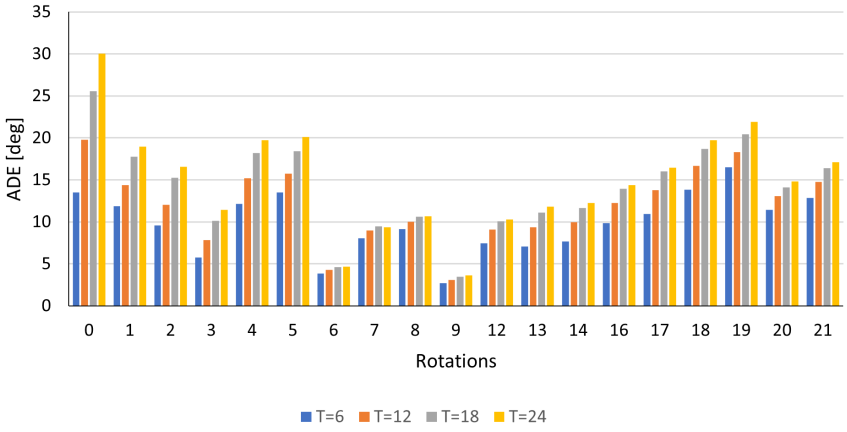

Figura 34: Errore euclideo medio nella predizione delle rotazioni 3D in un orizzonte di 6, 12, 18 e 24 passi.

il primo passo di predizione risulta meno accurato rispetto a quelli successivi. Nel dataset di test, le coordinate XYZ del root keypoint variano in range di ampiezza 6.9, 3.3 e 1.5 metri rispettivamente. Sebbene la maggior parte delle movimentazioni sia sul posto (fattore che influenza il calcolo dell'errore medio), avere un DE nell'ordine dei centimetri può dirsi un ottimo risultato.
Per quanto riguarda le rotazioni, il DE non fornisce un'informazione corretta, perché lo spazio della notazione asse-angolo non è euclideo. Occorre pertanto introdurre delle metriche alternative:

- in [25, 16] le rotazioni vengono espresse come terne di Eulero ($\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \alpha_3\}$) in modo da calcolare un *errore angolare* AE:

$$\text{AE}_t = \|\boldsymbol{\alpha}_{T_o+t} - \hat{\boldsymbol{\alpha}}_{T_o+t}\| \tag{35}$$

dove $\hat{\boldsymbol{\alpha}}$ rappresenta una stima rispetto al vettore vero $\boldsymbol{\alpha}$;

- in [2] viene suggerito l'utilizzo della *distanza angolare di giunto* (JAD, *joint angle distance*), intesa come la misura (in radianti) della rotazione necessaria ad allineare il sistema di riferimento predetto con quello vero. Per fare questo si riconducono le due rotazioni vera e stimata ad una rappresentazione matriciale ($\boldsymbol{R}$ e $\hat{\boldsymbol{R}}$), applicando poi $\check{\boldsymbol{R}}_t = \hat{\boldsymbol{R}}_t \boldsymbol{R}_t^{\mathrm{T}}$. $\check{\boldsymbol{R}}_t$ esprime una rotazione la cui misura è proprio la JAD.

Sia l'AE che la JAD si misurano in radianti. Il primo, tuttavia, soffre di due principali problemi: *(i)* dipende dalla sequenza di Eulero utilizzata per esprimere le rotazioni e *(ii)* per ogni rotazione esistono due possibili terne $\boldsymbol{\alpha}$. Questi fattori rendono questa metrica prona ad errori. Ne viene comunque riportato l'andamento temporale in Figura 35, mentre in Figura 36 sono esposti gli AE mediati su diversi orizzonti (la sequenza scelta è la ZXY con riferimento alla Figura 25). L'andamento della JAD è invece presentato in Figura 37, accompagnato dai valori mediati in Figura 38.
A differenza dei risultati 2D in Sezione 4.4, qui gli intervalli di variazione delle traiettorie sono caratteristici di ciascun giunto: sbagliare di pochi gradi nella rotazione del ginocchio, ad esempio, è meno grave di commettere lo stesso errore quantitativo in un giunto della colonna vertebrale, assai meno variante. I grafici confermano questa tendenza. L'orientazione assoluta (indice 0), infatti, è quella che viene predetta nel modo meno accurato. Questo fatto non dovrebbe sorprendere, perchè essa dipende da fattori esterni, quali l'intenzione della persona o la presenza di ostacoli, variabili di cui non si tiene conto in questo lavoro. Seguono ginocchia (indici 4, 5), anche (indici 1, 2), gomiti (indici 18, 19) e polsi (indici 20, 21). Le predizioni migliori sono quelle sulla colonna vertebrale (indici 3, 6, 9) e sulle caviglie (indici 7, 8).
Nel complesso, le JAD si mantengono a livelli accettabili, non superando mai i 20 gradi prima del decimo passo di predizione.

4.6 APPLICAZIONE DEI VINCOLI

Prima di combinare le traiettorie 3D predette al fine di ottenere i movimenti umani finali, si applica il procedimento di applicazione dei vincoli anatomici descritto in Sezione 3.4. Si rammenta che le traiettorie relative alla traslazione del corpo nello spazio e alla sua

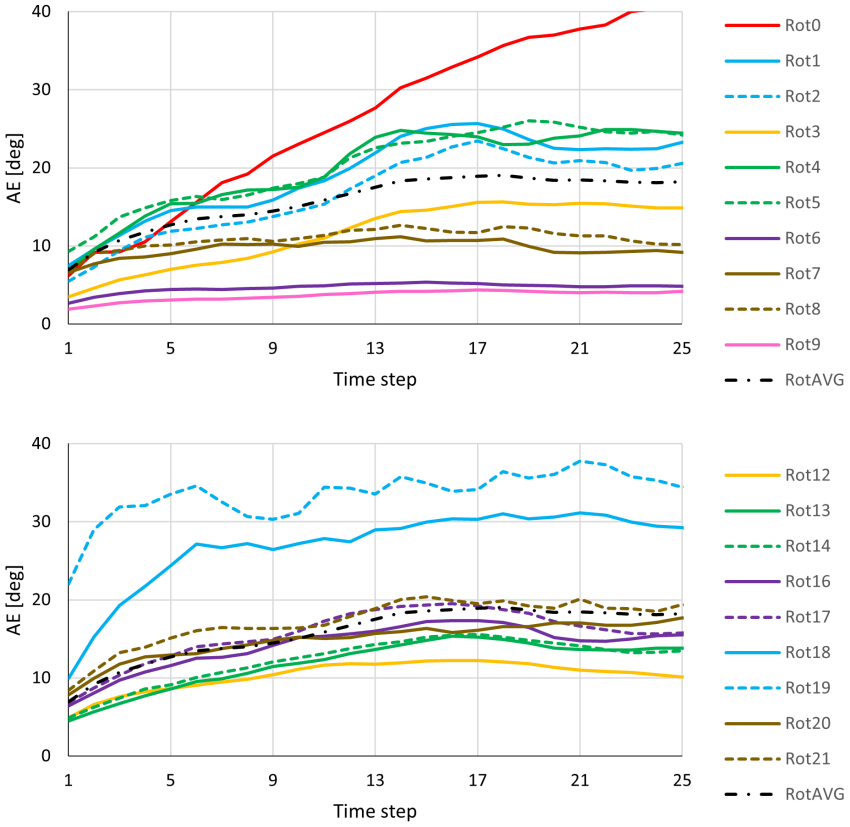

Figura 35: Errore angolare nella predizione delle rotazioni 3D.

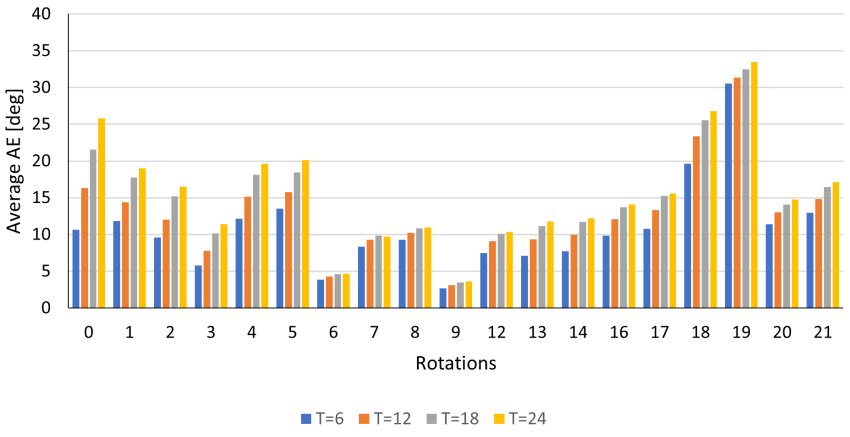

Figura 36: Errore angolare medio nella predizione delle rotazioni 3D in un orizzonte di 6, 12, 18 e 24 passi.

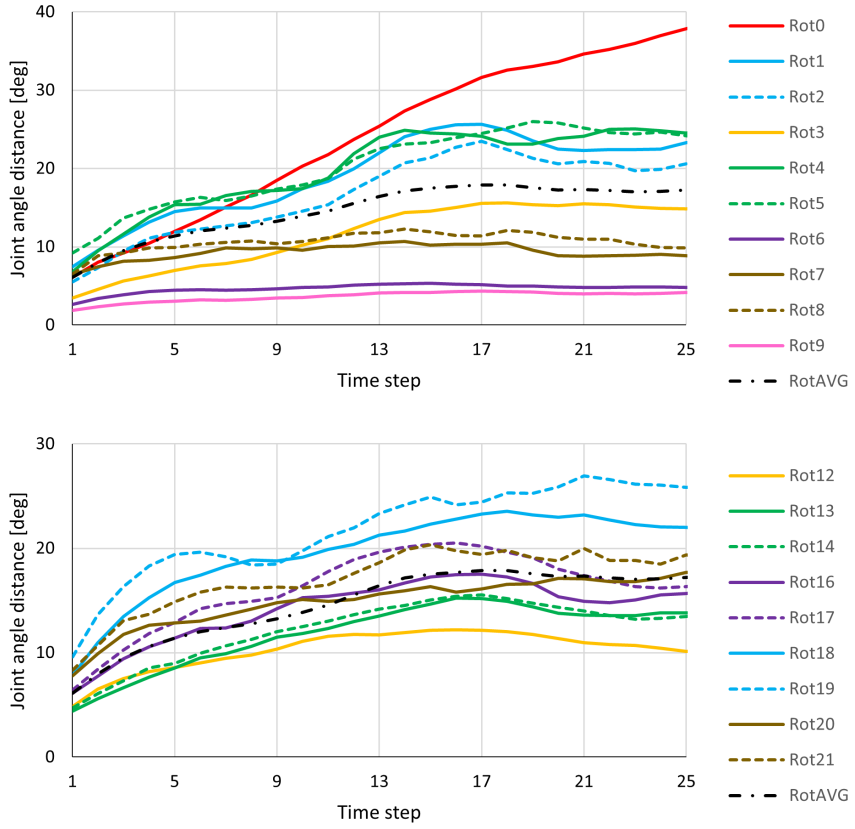

Figura 37: Distanza angolare di giunto nella predizione delle rotazioni 3D.

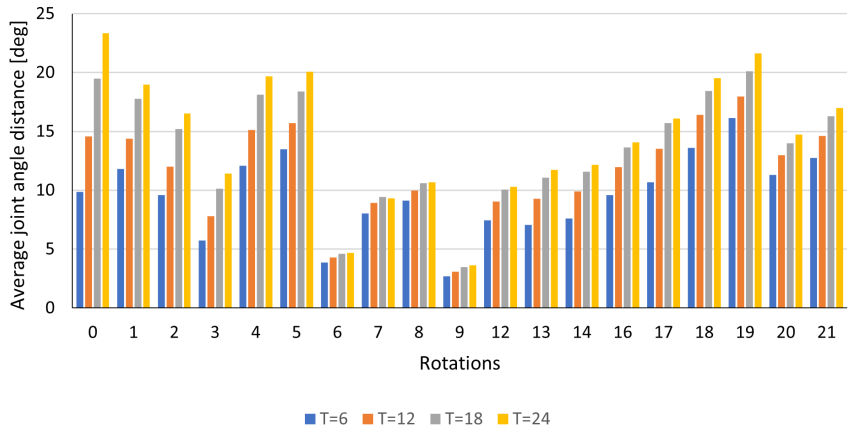

Figura 38: Media distanza angolare di giunto nella predizione delle rotazioni 3D in un orizzonte di 6, 12, 18 e 24 passi.

Indice giunto	Nome giunto	Traiettorie fuori range	Punti fuori range	Totale (deg)	Media (deg)
1	anca sx	1	13	2.14	0.16
2	anca dx	0	0	0.00	-
3	col. vert. 1	0	0	0.00	-
4	ginocchio sx	2	9	11.21	1.25
5	ginocchio dx	1	3	0.82	0.27
6	col. vert. 2	0	0	0.00	-
7	caviglia sx	0	0	0.00	-
8	caviglia dx	0	0	0.00	-
9	col. vert. 3	0	0	0.00	-
12	collo	0	0	0.00	-
13	clavicola sx	0	0	0.00	-
14	clavicola dx	0	0	0.00	-
16	spalla sx	0	0	0.00	-
17	spalla dx	0	0	0.00	-
18	gomito sx	0	0	0.00	-
19	gomito dx	1	2	0.59	0.30
20	polso sx	2	57	153.47	2.69
21	polso dx	4	91	202.80	2.22

Tabella 3: Risultato dell'applicazione dei vincoli anatomici.

orientazione non subiscono questo processo, in quanto non risentono di alcun limite intrinseco.

Dato un generico giunto j, ogni traiettoria 3D associata ($\hat{\boldsymbol{X}}^{f}_{j,T_o+1:T_s}$) ha un certo numero di punti che risultano essere fuori dai limiti consentiti. La correzione di queste dislocazioni permette di generare una nuova traiettoria $\tilde{\boldsymbol{X}}^{f}_{j,T_o+1:T_s}$ più simile possibile a quella originale, ma garantendo che i limiti anatomici vengano sempre rispettati. In Tabella 3 sono riportati, per ogni giunto, il numero di traiettorie e di punti che non rispettano i vincoli oltre che l'ampiezza totale e media dell'errore introdotto (in termini di JAD). Data la limitata diversità ottenuta nelle predizioni multi-futuro, sono presentati i risultati solo per il primo scenario futuro ($f = 1$), sapendo che questi sono abbastanza rappre-

sentativi anche degli altri $F_J - 1$ futuri predetti per ogni traiettoria.
Come si può notare, il sistema di applicazione dei vincoli entra in gioco un numero di volte relativamente ridotto: massimo 4 per il polso destro (si ricorda che i motion files di test sono 293 e che la sequenza predetta ha una lunghezza di 48 frame). In più, l'entità della modifica necessaria risulta essere al massimo di pochi gradi, con le peggiori condizioni sempre in corrispondenza dei polsi. Per la maggior parte dei giunti, comunque, le traiettorie predette soddisfano già i limiti anatomici.
Le possibili ragioni di questo comportamento sono molteplici. Innanzitutto, il dataset usato per i test potrebbe essere tale da non richiedere una stringente imposizione dei vincoli. Inoltre, la scelta di utilizzare una tecnica di riduzione della dimensionalità sulle traiettorie 3D osservate forza alcuni giunti a variare entro uno spazio più ristretto di quello iniziale. Un altro fattore che probabilmente contribuisce è la mappatura delle traiettorie 2D (ridotte dal 3D con la PCA) in una griglia ben definita. I confini di quest'ultima, di fatto, costituiscono un limite ulteriore alle possibilità di variazione dei parametri dei giunti.
Nessuno dei motivi sopra esposti dovrebbe però consigliare l'eliminazione o la limitazione di questa fase, perché in nessun altro punto del sistema viene definito alcun tipo di vincolo esplicito sulle pose umane ammissibili. In particolari contesti, pertanto, l'assenza di questo stadio potrebbe comportare delle conseguenze negative.

4.7 RISULTATI FINALI

Una volta ottenute le traiettorie 3D di giunto vincolate, è possibile combinarle in modo da ottenere la predizione multi-futuro richiesta sui movimenti di un essere umano. Il metodo praticato segue le indicazioni fornite in Sezione 3.5.
Per prima cosa, va detto che la traslazione globale del corpo è già stata valutata nella Sezione 4.5 attraverso il displacement error in 3D. Nel seguito, ci si focalizzerà sulla predizione delle posture corrette, informazione che è indipendente dalla collocazione del soggetto nello spazio.
Le metriche disponibili in letteratura per la valutazione dei movimenti umani sono molteplici [2, 22]:

- MJAD (*mean joint angle distance*), ovvero la media delle JAD ottenute per ognuno dei J giunti più l'orientazione globale:

$$\text{MJAD}_t = \frac{1}{J+1} \sum_j \text{JAD}_{j,t} \tag{36}$$

(espressa in radianti);

- MPJPE (*mean per joint position error*), definito come la distanza euclidea media tra le posizioni 3D predette ($\boldsymbol{p}_k$) e quelle vere ($\hat{\boldsymbol{p}}_k$) degli N_{kp} keypoint del corpo umano (giunti, root keypoint e organi terminali):

$$\text{MPJPE}_t = \frac{1}{N_{kp}} \sum_k \|\boldsymbol{p}_{k,t} - \hat{\boldsymbol{p}}_{k,t}\| \tag{37}$$

(espresso in metri);

- PCK (*percentage of correct keypoints*), la quale indica la quota parte di posizioni di keypoints predette in maniera "corretta", ovvero entro una distanza massima (soglia) ρ:

$$\text{PCK}_{\rho,t} = \frac{1}{N_{kp}} \sum_k \tau\left(\boldsymbol{p}_{k,t}, \hat{\boldsymbol{p}}_{k,t}, \rho\right) \tag{38}$$

con:

$$\tau\left(\boldsymbol{p}_{k,t}, \hat{\boldsymbol{p}}_{k,t}, \rho\right) = \begin{cases} 1 & \text{se } \|\boldsymbol{p}_{k,t} - \hat{\boldsymbol{p}}_{k,t}\| \leq \rho \\ 0 & \text{altrimenti} \end{cases} \tag{39}$$

A differenza delle due metriche precedenti questa non indica un errore, bensì un'accuratezza. Pertanto, migliori performance sono associate ad alti PCK (possibilmente 100%).

Come riportato in [25], un buon sistema di predizione del movimento umano dovrebbe quantomeno avere prestazioni quantitative superiori rispetto al modello zero-velocity, il quale semplicemente replica l'ultima posa osservata per tutto l'orizzonte di predizione. Nei grafici successivi, pertanto, si segnano gli errori definiti in precedenza sia per il modello implementato che per lo zero-velocity, in modo da avere un metro di giudizio affidabile. Si riportano i valori istantanei e medi di MJAD (Figure 39 e 40), MPJPE (Figure 41 e 42) e PCK con soglie ρ di 5, 10 e 20 centimetri (Figure 43 e 44).

Dal confronto tra l'algoritmo proposto e il modello zero-velocity, si possono trarre alcune considerazioni. Innanzitutto, si nota che zero-velocity restituisce risultati migliori al primo passo di predizione.

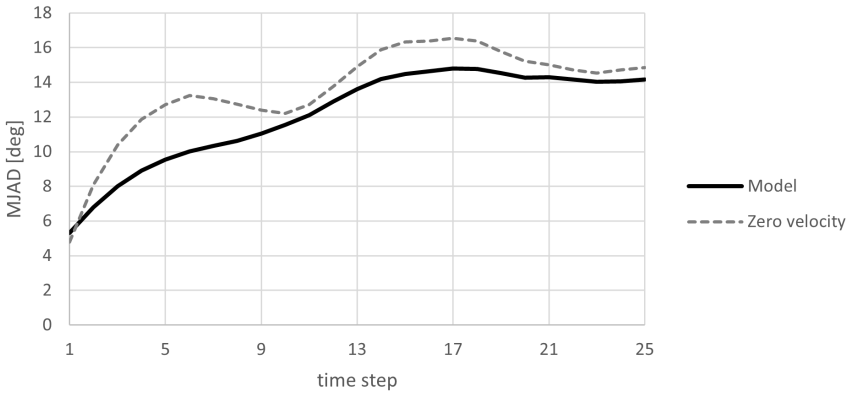

Figura 39: MJAD nella predizione dei movimenti umani.

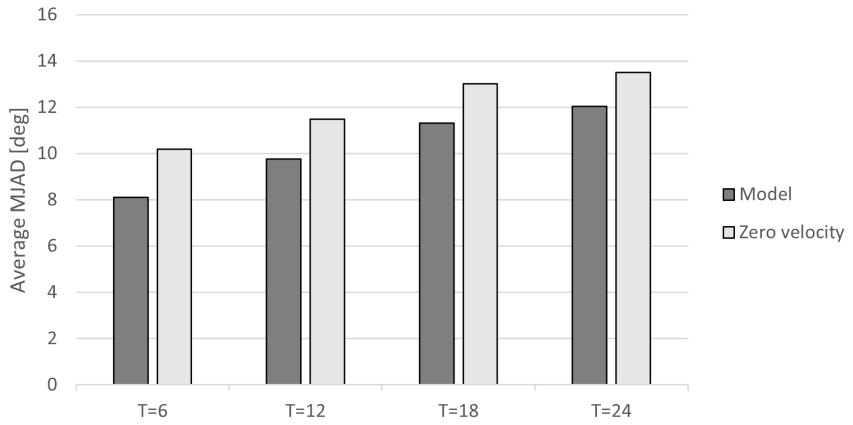

Figura 40: MJAD medio nella predizione dei movimenti umani in un orizzonte di 6, 12, 18 e 24 passi.

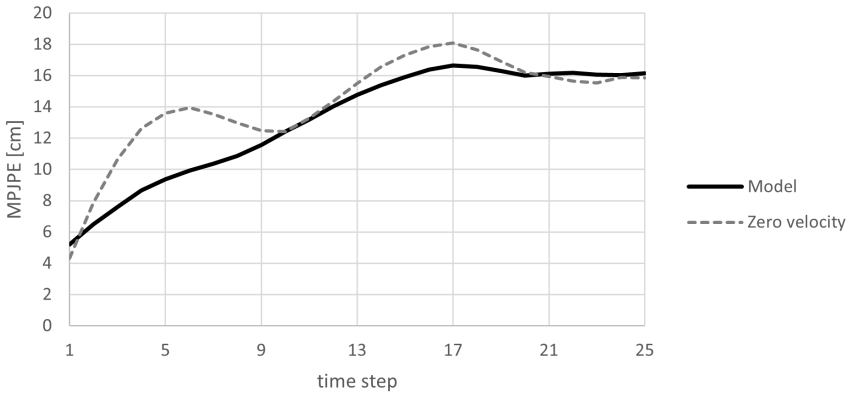

Figura 41: MPJPE nella predizione dei movimenti umani.

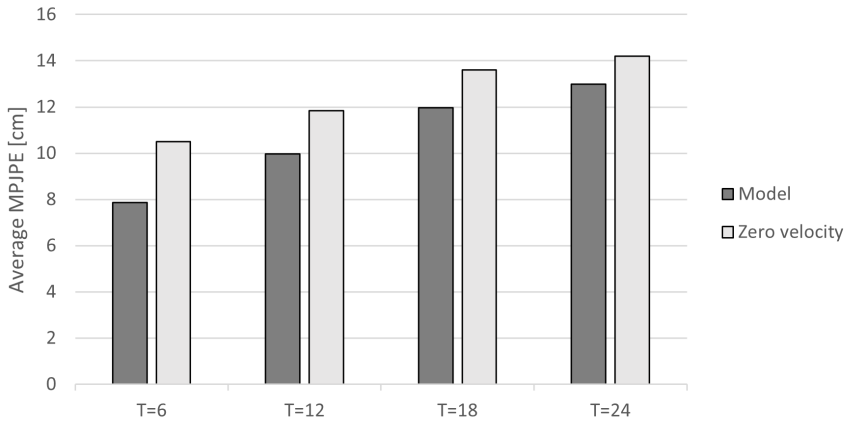

Figura 42: MPJPE nella predizione dei movimenti umani in un orizzonte di 6, 12, 18 e 24 passi.

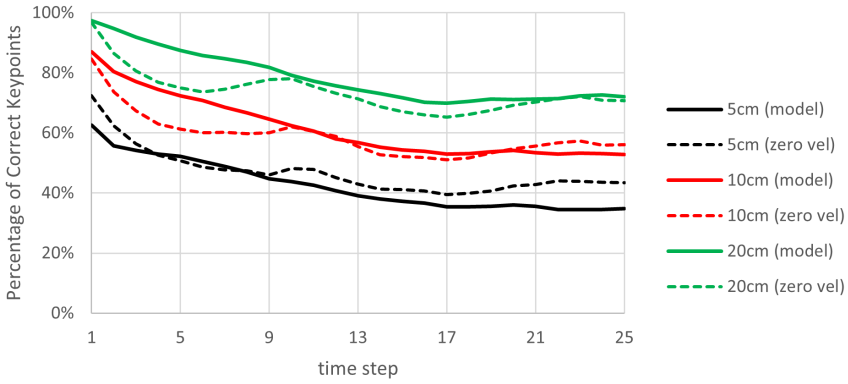

Figura 43: PCK nella predizione dei movimenti umani con soglie di 5, 10 e 20 cm.

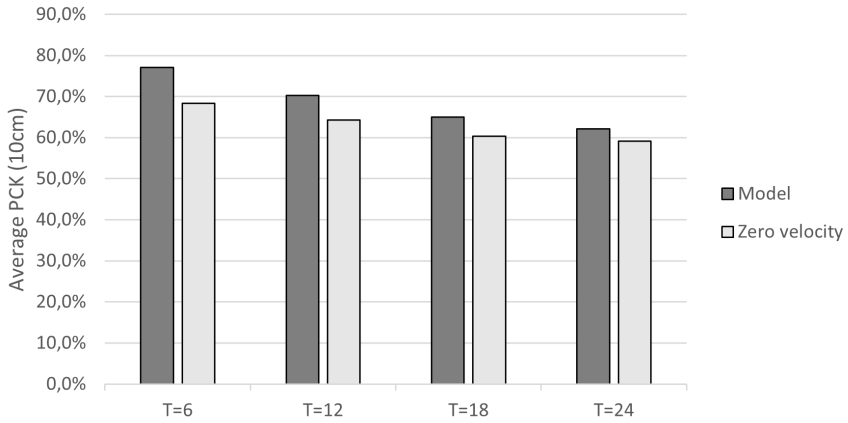

Figura 44: PCK medio nella predizione dei movimenti umani con soglia di 10 cm in un orizzonte di 6, 12, 18 e 24 passi.

Evidentemente, il sistema implementato soffre di un'eccessiva discontinuità al primo istante di predizione (problema descritto in [25]). Una possibile soluzione potrebbe essere quella di operare con le velocità al posto delle posizioni, approccio che però richiede di calcolare le derivate dei dati AMASS, con tutti gli svantaggi che ne derivano (soprattutto in termini di introduzione di rumore). Dal secondo passo in poi, comunque, il modello offre dei risultati accettabili. Sino a $t \approx 10$ (2 secondi), gli errori sono sensibilmente più bassi rispetto allo zero-velocity, e anche aumentando l'orizzonte temporale i valori restano buoni.
In termini assoluti, comunque, si ottengono errori accettabili. Nei primi 2 secondi la MJAD si mantiene entro i 10 gradi, il MPJPE rimane nell'ordine dei 10 centimetri e il PCK supera l'80% con una soglia di 20 cm (si ricorda che lo scheletro SMPL sfruttato in questo lavoro ha le dimensioni fisiche di un essere umano di corporatura media alto 170 cm).
Analizzando gli output del modello, si realizza che la maggior fonte di errori è data da movimenti di natura non periodica. In particolare, le più grandi deviazioni si hanno in corrispondenza di sequenze in cui il soggetto compie una "transizione", passando ad esempio dal camminare al correre oppure dal camminare al sedersi. Nella realtà queste transizioni avvengono per una motivazione, legata alle intenzioni del soggetto (ad esempio la presenza di un oggetto a terra potrebbe spingere la persona a chinarsi per raccoglierlo) e/o all'ambiente in cui egli è collocato (la presenza di un muro obbliga il soggetto a cambiare direzione ed eventualmente rallentare). Questi tipi di movimentazione sono ovviamente più difficili da anticipare per un sistema automatico, soprattutto in assenza delle informazioni sopra citate. Come già scritto in precedenza, questo lavoro non prende in considerazione questi fattori, pertanto le criticità riscontrate sono difficili da valutare negativamente.

CONCLUSIONI

L'obiettivo di questa tesi era lo sviluppo di un algoritmo per l'anticipazione dei movimenti futuri di un essere umano, attività estremamente complessa poiché il moto delle persone dipende da un numero molto grande di fattori, legati sia alla natura del corpo umano che al contesto circostante. L'ambito della predizione del movimento ha suscitato un ampio interesse specialmente negli ultimi anni, grazie soprattutto all'evoluzione delle tecniche di machine learning, le quali consentono di trattare il problema con un approccio diverso, che non ne richiede una modellazione approfondita ed esplicita.
Per svolgere il compito richiesto, si è deciso di suddividere la catena cinematica dello scheletro umano in ognuno dei suoi giunti, per poi effettuare una predizione della traiettoria per ciascuno di essi in maniera indipendente dagli altri. Il principale vantaggio di questa strada è la riduzione della complessità del problema, dato che il numero di parametri descrittivi di una posa umana è molto maggiore rispetto a quelli di un punto nello spazio.
L'approccio adottato ha reso necessaria l'introduzione di un sistema che fosse in grado di fornire delle predizioni accurate su queste traiettorie. La scelta è ricaduta su Multiverse, un modello basato sulle reti neurali che ha già dimostrato la sua affidabilità in contesti di tracciamento di percorsi descritti da individui all'interno di scene bidimensionali. Multiverse risolve il problema della predizione delle traiettorie mappandole in una griglia 2D, e generando ad ogni istante una heatmap che indica le probabilità di occupazione di ciascuna cella. Questa stima viene migliorata computando, per ogni cella, un offset rispetto al centro. Le heatmap, nello specifico, consentono di elaborare una predizione multi-futuro.
L'utilizzo di una struttura di questo tipo, adatta a trattare dati in 2D, ha richiesto l'inserimento di uno stadio di riduzione della dimensionalità dell'input in 3D, accettando in questo modo una deteriorazione dell'informazione originale. La tecnica applicata è l'analisi delle componenti principali, che assolve a questa mansione assicurando le minime alterazioni in tal senso. Dai risultati riscontrati, si è visto come l'impiego della PCA comporti delle perdite accettabili, anche per via del fatto

che molti giunti del corpo umano effettivamente variano in uno spazio ben descrivibile anche con sole due componenti.
Le traiettorie di giunto non sono state definite nello spazio metrico (ad eccezione della traslazione globale), bensì in quello delle rotazioni, in particolare nel dominio della notazione asse-angolo. In questo modo si è notevolmente semplificato il compito, perché i parametri rotazionali delle pose sono indipendenti dalle dimensioni fisiche di ogni persona considerata. Inoltre, anche l'espressione matematica dei vincoli anatomici si è rivelata molto più agevole.
I risultati ottenuti nella fase di test sono promettenti, soprattutto se si considera che due fattori di primaria importanza, ovvero contesto e intenzioni del soggetto, sono stati completamente trascurati nello sviluppo dell'algoritmo. L'aggiunta di queste indicazioni renderebbe gli ingressi più specifici e informativi, incrementando le capacità di anticipazione del modello per movimenti anche molto complessi e varianti.
Un inconveniente che si è riscontrato è la difficoltà da parte del sistema di effettuare delle predizioni multi-futuro realmente distinte tra loro. Un allenamento più intensivo del modello, con dataset più variegati, potrebbe favorire un rafforzamento delle facoltà di differenziazione in fase di previsione. Anche in questa circostanza, comunque, non sono da sottovalutare i potenziali benefici che si possono trarre dall'introduzione dei dati ambientali, i quali donerebbero al sistema una maggiore consapevolezza del contesto in cui andrebbe ad operare.
In conclusione, l'approccio utilizzato in questo lavoro si è rivelato efficace nel compito dell'anticipazione del movimento umano. In futuro, i perfezionamenti sopra descritti potrebbero migliorare ulteriormente le prestazioni.

BIBLIOGRAFIA

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Gregory S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian J. Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Józefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Gordon Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul A. Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda B. Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *CoRR*, abs/1603.04467, 2016.

[2] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. Structured prediction helps 3d human motion modelling. *CoRR*, abs/1910.09070, 2019.

[3] Matthew Brand and Aaron Hertzmann. Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 183–192, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1581132085. doi: 10.1145/344779.344865.

[4] Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. Massive exploration of neural machine translation architectures. *CoRR*, abs/1703.03906, 2017.

[5] Judith Bütepage, Michael J. Black, Danica Kragic, and Hedvig Kjellström. Deep representation learning for human motion prediction and classification. *CoRR*, abs/1702.07486, 2017.

[6] Scott L. Delp, Frank C. Anderson, Allison S. Arnold, Peter Loan, Ayman Habib, Chand T. John, Eran Guendelman, and Darryl G. Thelen. Opensim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Trans. Biomed. Engineering*, 54(11):1940–1950, 2007.

[7] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning, 2016.

[8] Katerina Fragkiadaki, Sergey Levine, and Jitendra Malik. Recurrent network models for kinematic tracking. *CoRR*, abs/1508.00271, 2015.

[9] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. Learning human motion models for long-term predictions. *CoRR*, abs/1704.02827, 2017.

[10] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[11] F. Sebastin Grassia. Practical parameterization of rotations using the exponential map. *J. Graph. Tools*, 3(3):29–48, mar 1998. ISSN 1086-7651. doi: 10.1080/10867651.1998.10487493.

[12] Martin T. Hagan, Howard B. Demuth, Mark H. Beale, and Orlando De Jesús. *Neural Network Design (2nd Edition)*. 2014.

[13] Fei Han, Brian Reily, William A. Hoff, and Hao Zhang. Space-time representation of people based on 3d skeletal data: A review. *CoRR*, abs/1601.01006, 2016.

[14] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585 (7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2.

[15] Kelli D. Humbird, J. Luc Peterson, and Ryan G. McClarren. Predicting the time-evolution of multi-physics systems with sequence-to-sequence models. *CoRR*, abs/1811.05852, 2018.

[16] Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. *CoRR*, abs/1511.05298, 2015.

[17] Hema Swetha Koppula and Ashutosh Saxena. Anticipating human activities for reactive robotic response. In *2013 IEEE/RSJ*

International Conference on Intelligent Robots and Systems, pages 2071–2071, 2013. doi: 10.1109/IROS.2013.6696634.

[18] Weilun Lao, Jungong Han, and Peter H.n. De With. Automatic video-based human motion analyzer for consumer surveillance system. *IEEE Transactions on Consumer Electronics*, 55(2):591–598, 2009. doi: 10.1109/TCE.2009.5174427.

[19] Andreas M. Lehrmann, Peter V. Gehler, and Sebastian Nowozin. A non-parametric bayesian network prior of human pose. In *2013 IEEE International Conference on Computer Vision*, pages 1281–1288, 2013. doi: 10.1109/ICCV.2013.162.

[20] Junwei Liang, Lu Jiang, Kevin P. Murphy, Ting Yu, and Alexander G. Hauptmann. The garden of forking paths: Towards multi-future trajectory prediction. *CoRR*, abs/1912.06445, 2019.

[21] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, October 2015.

[22] Kedi Lyu, Haipeng Chen, Zhenguang Liu, Beiqi Zhang, and Ruili Wang. 3d human motion prediction: A survey. *Neurocomputing*, 489:345–365, 2022.

[23] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: archive of motion capture as surface shapes. *CoRR*, abs/1904.03278, 2019.

[24] Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The kit whole-body human motion database. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 329–336, 2015. doi: 10.1109/ICAR.2015.7251476.

[25] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. *CoRR*, abs/1705.02445, 2017.

[26] Meinard Müller, Tido Röder, Michael Clausen, Bernhard Eberhardt, Björn Krüger, and Andreas Weber. Mocap database hdm05. *Institut für Informatik II, Universität Bonn*, 2(7), 2007.

[27] Brian Paden, Michal Cáp, Sze Zheng Yong, Dmitry S. Yershov, and Emilio Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *CoRR*, abs/1604.07446, 2016.

[28] Jae Sung Park and Dinesh Manocha. HMPO: human motion prediction in occluded environments for safe motion planning. *CoRR*, abs/2006.00424, 2020.

[29] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985, 2019.

[30] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[31] Nkolika J Peter, Hilary I Okagbue, Emmanuela CM Obasi, and Adedotun O Akinola. Review on the application of artificial neural networks in real estate valuation. *International Journal*, 9 (3), 2020.

[32] Mark Richardson. Principal component analysis. 2009.

[33] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), November 2017.

[34] Elham S. Salama, Reda A. El-Khoribi, Mahmoud E. Shoman, and Mohamed A. Wahby Shalaby. A 3d-convolutional neural network framework with ensemble learning techniques for multi-modal emotion recognition. *Egyptian Informatics Journal*, 22(2):167–176, 2021. ISSN 1110-8665.

[35] István Sárándi, Timm Linder, Kai Oliver Arras, and Bastian Leibe. How robust is 3d human pose estimation to occlusion? *CoRR*, abs/1808.09316, 2018.

[36] Paul Schydlo, Mirko Rakovic, Lorenzo Jamone, and José Santos-Victor. Anticipation in human-robot cooperation: A recurrent neural network approach for multiple action sequences prediction. *CoRR*, abs/1802.10503, 2018.

[37] Elnaz Siami-Irdemoosa and Saeid R Dindarloo. Prediction of fuel consumption of mining dump trucks: A neural networks approach. *Applied Energy*, 151:77–84, 2015.

[38] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[39] Stefano Tortora, Stefano Michieletto, Francesca Stival, and Emanuele Menegatti. Fast human motion prediction for human-robot collaboration with wearable interfaces. *CoRR*, abs/1905.11734, 2019.

[40] Nikolaus F. Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of Vision*, 2(5):2–2, 09 2002. ISSN 1534-7362. doi: 10.1167/2.5.2.

[41] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[42] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008. doi: 10.1109/TPAMI.2007.1167.

[43] Lei Wang, Yan Wang, Deng Cai, Dongxiang Zhang, and Xiaojiang Liu. Translating a math word problem to an expression tree. *CoRR*, abs/1811.05632, 2018.

[44] Yunbo Wang, Lu Jiang, Ming-Hsuan Yang, Li-Jia Li, Mingsheng Long, and Li Fei-Fei. Eidetic 3d lstm: A model for video prediction and beyond. In *ICLR*, 2019.

[45] Sam Wiseman and Alexander M. Rush. Sequence-to-sequence learning as beam-search optimization. *CoRR*, abs/1606.02960, 2016.

[46] Ge Wu, Peter R Cavanagh, et al. Isb recommendations for standardization in the reporting of kinematic data. *Journal of biomechanics*, 28(10):1257–1262, 1995.

[47] Ge Wu, Sorin Siegler, Paul Allard, Chris Kirtley, Alberto Leardini, Dieter Rosenbaum, Mike Whittle, Darryl D D'Lima, Luca Cristofolini, Hartmut Witte, et al. Isb recommendation on definitions of joint coordinate system of various joints for the reporting of human joint motion-part i: ankle, hip, and spine. *Journal of biomechanics*, 35(4):543–548, 2002.

[48] Ge Wu, Frans CT Van der Helm, HEJ DirkJan Veeger, Mohsen Makhsous, Peter Van Roy, Carolyn Anglin, Jochem Nagels, Andrew R Karduna, Kevin McQuade, Xuguang Wang, et al. Isb recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion-part ii: shoulder, elbow, wrist and hand. *Journal of biomechanics*, 38(5): 981–992, 2005.

[49] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 2015.

[50] Xsens. *MVN User Manual*, 2021.

[51] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.