Università degli Studi di Modena e Reggio Emilia

DIPARTIMENTO DI SCIENZE E METODI DELL'INGEGNERIA

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA MECCATRONICA

TESI DI LAUREA MAGISTRALE

Valutazione di Fattibilità e Pipeline End-to-End per l'Integrazione di Robot Umanoidi nei Processi Industriali

Dalla Stima del Movimento Umano da Video 2D al Retargeting e Proof of Concept in Simulazione Dinamica su Unitree G1

Candidato:
Ardolino Alessandro

Relatore:
Prof. Riccardo Rubini

Correlatori:
Dir. Ing. Lorenzo Vezzani
Ing. Bruno Ferrarini

Anno Accademico 2025–2026

Sommario

Parole chiave: Robotica Umanoide, Visione Artificiale, Kinematic Retargeting, Simulazione Dinamica, Unitree G1.

L'integrazione di robot umanoidi nei contesti industriali rappresenta una delle frontiere più promettenti per superare la rigidità dell'automazione tradizionale e abilitare processi produttivi flessibili in ambienti originariamente concepiti per l'operatore umano. Il presente lavoro di tesi magistrale, sviluppato in collaborazione con il reparto di Ricerca e Sviluppo di Nexion Group, affronta la sfida di trasferire abilità motorie umane verso un robot umanoide compatto (Unitree G1) in modo rapido e non invasivo, sfruttando semplici registrazioni video 2D al posto dei costosi e complessi sistemi di motion capture tradizionali.

Il percorso di ricerca ha preso avvio da una fase sperimentale preliminare volta a testare sistemi markerless multi-camera basati su visione classica, della quale sono stati analizzati i limiti pratici in ambiente di officina e le criticità cinematiche nel generare movimenti compatibili con la struttura del robot. A partire da questi risultati, è stata sviluppata una pipeline software end-to-end che impiega moderni modelli di visione monoculare per ricostruire il movimento tridimensionale da singoli video RGB e un modulo di retargeting cinematico per adattare le traiettorie umane ai gradi di libertà del robot, garantendo il rispetto dei limiti articolari e la stabilità dell'appoggio al suolo.

Le traiettorie generate sono state successivamente validate all'interno di un ambiente di simulazione fisica e dinamica, intavolando un Proof of Concept per il controllo del robot tramite tecniche di apprendimento e conducendo un'analisi critica sui limiti dell'imitazione puramente antropomorfa rispetto alle piene potenzialità meccaniche del sistema. Infine, l'intera indagine metodologica e bibliografica è stata organizzata mediante un sistema strutturato di gestione della conoscenza, concepito come asset duraturo per il trasferimento tecnologico all'interno del reparto R&D aziendale.

Indice

Sommario **i**

1 Introduzione e Inquadramento nel Contesto R&D Industriale **1**
1.1 Il Paradigma dell'Umanoide nei Processi Produttivi 1
1.1.1 Limiti dell'automazione rigida e flessibilità antropomorfa 1
1.1.2 Il contesto industriale: Nexion Group e la visione dell'R&D 2
1.1.3 Obiettivi e perimetro del lavoro di tesi 2
1.2 Dal Mocap Invasivo alla Visione Computazionale Non Invasiva 3
1.2.1 Criticità operative dei sistemi mocap tradizionali in fabbrica 3
1.2.2 L'opportunità dei modelli di visione per il trasferimento del moto . 3
1.2.3 La piattaforma target: Unitree G1 4
1.3 Panoramica della Pipeline Proposta e Metodologia di Ricerca 4
1.3.1 Schema logico della pipeline end-to-end 4
1.3.2 Metodologia di ricerca aumentata e trasferimento del know-how (LLM-Wiki) 6
1.3.3 Infrastruttura hardware di calcolo e apparati di acquisizione 6
1.4 Struttura dell'Elaborato 7

2 Studio Sperimentale su FreeMoCap: Setup, Test e Analisi dei Colli di Bottiglia **9**
2.1 Architettura e Setup del Banco di Prova Multi-Camera 9
2.1.1 Configurazione dell'ambiente software dedicato e dell'array di telecamere 11
2.1.2 Calibrazione intrinseca/estrinseca con tavola ChArUco e piano del pavimento 13
2.1.3 Pipeline algoritmica di triangolazione DLT e reiezione degli outlier . 16
2.2 Risultati Sperimentali e Limiti Operativi in Ambiente Reale 17
2.2.1 Sensibilità alle condizioni ambientali, falsi positivi e perdita di tracciamento 17
2.2.2 Instabilità nella definizione del piano del suolo ($Z = 0$) 19
2.2.3 Collo di bottiglia computazionale su CPU e instabilità di sistema . 20

2.3 L'Ostacolo Critico: Incompatibilità Cinematica e la Sfida del Retargeting . 21
2.3.1 Mismatch topologico tra landmark cartesiani e requisiti dei motori di retargeting . . . 21
2.3.2 Varianza delle lunghezze ossee, assenza di vincoli $SO(3)$ e shoulder popping . . . 22
2.3.3 Motivazione del cambio di paradigma tecnologico . . . 23

3 Architettura Software, Gestione delle Dipendenze ed Ecosistema Modulare 25
3.1 Risoluzione dei Conflitti Software e Strategia degli Ambienti Isolati . . . 25
3.1.1 Il problema del Dependency Hell nella robotica moderna . . . 26
3.1.2 Ambienti virtuali standard (`venv`) vs ecosistemi Conda: gestione dei pacchetti e file di configurazione (`.txt` vs `.yaml`) . . . 27
3.1.3 Mappatura e isolamento dei 4 ambienti Conda dedicati . . . 29
3.1.4 Gestione dei conflitti tra driver CUDA, versioni PyTorch e OpenUSD 30
3.2 Protocolli di Interscambio Dati e Pipeline Orchestration . . . 31
3.2.1 Flusso di interscambio dati end-to-end e formati di comunicazione . 31
3.2.2 Manutenibilità, modularità e aggiornabilità dei singoli stadi . . . 32
3.2.3 Riproducibilità scientifica ed esportazione delle configurazioni `environment.yaml` . . . 32
3.3 Quadro di Sintesi dell'Ecosistema Software . . . 34

4 Visione Monoculare Neurale e Kinematic Retargeting con GEM-X e SOMA 35
4.1 Estrazione del Movimento con GEM-X . . . 35
4.1.1 Acquisizione Video 2D e Scenario Operativo . . . 36
4.1.2 Architettura a Stadi: YOLOX, ByteTrack e ViTPose a 77 Keypoint 37
4.1.3 GEM Core Temporal Transformer e Separazione del Movimento . . 39
4.1.4 Finestra Temporale, Dinamica del Gesto e Strategia Operativa . . . 40
4.1.5 Risoluzione dei Difetti del BVH: Rigidità dello Scheletro e Continuità del Moto . . . 41
4.2 Integrazione GEM-X / SOMA e Retargeting su Unitree G1 . . . 41
4.3 Validazione Sperimentale e Ispezione delle Traiettorie . . . 43
4.3.1 Conversione Standalone e Ispezione con `bvhtocsv` . . . 43
4.3.2 Prove con Variazione della Prospettiva e Registrazioni a Mano Libera 44
4.3.3 Prove di Manipolazione con Utensili e Confronto di Risoluzione . . 45
4.3.4 Prove su Posture Chine e Ambiente Reale di Officina . . . 46
4.3.5 Sintesi dei Risultati e Ponte verso la Simulazione Dinamica . . . 46

5 Proof of Concept in Simulazione Dinamica (Isaac Lab) e Analisi Critica dell'Imitazione **49**
5.1 Introduzione agli Ambienti di Simulazione: Isaac Sim e Isaac Lab 49
5.1.1 Panoramica: Motore Fisico vs Framework RL 49
5.1.2 Vantaggi sui Competitor e Riduzione del Sim-to-Real Gap 50
5.1.3 Le Sfide dell'Adozione: Strumento all'Avanguardia ma Acerbo . . . 51
5.2 Motivazioni e Scopo del Proof of Concept (PoC) 52
5.2.1 La Scelta di Strumenti Open-Source State-of-the-Art 52
5.2.2 Perimetro di Utilizzo: Ricerca e Rimodulazione di Progetti Esistenti 52
5.3 Dinamiche di Apprendimento: RL Massivo e Domain Randomization . . . 53
5.3.1 Parallelizzazione su GPU e Algoritmo PPO 53
5.3.2 Domain Randomization come Ponte verso la Realtà 53
5.4 Configurazione Pratica del Task per l'Unitree G1 54
5.4.1 Asset USD, Inerzie e Collisioni 54
5.4.2 Paradigmi Architetturali: Manager-Based vs Direct Workflow . . . 54
5.4.3 Spazio delle Osservazioni, Azioni e Reward Engineering 56
5.5 Sviluppo del Parser CSV e Integrazione in Isaac Lab 56
5.5.1 Generatore di Comandi e Mappatura Cinematica 56
5.5.2 Decorrelazione Spaziotemporale (Resampling) per il Reinforcement Learning 57
5.5.3 Funzioni di Reward basate su Kernel Esponenziale e Termini di Fluidità 57
5.5.4 Il rischio di Reward Hacking, Tuning dei Pesi e Validazione Visiva . 58
5.6 Esecuzione dell'Addestramento e Validazione del Proof of Concept 60
5.6.1 Configurazione Plug-and-Play e Lancio su Terreni Paralleli 60
5.6.2 Telemetria da Riga di Comando ed Evoluzione dell'Addestramento . 62
5.6.3 Monitoraggio Grafico e Telemetria Avanzata (TensorBoard) 64
5.7 Analisi Critica: I Limiti dell'Imitazione Umana Pura (Il Collo di Bottiglia Antropomorfo) 66
5.7.1 Il bias del dato umano e la compressione dei range articolari 66
5.7.2 Le capacità meccaniche inespresse del G1: mobilità del torso e range articolari 66
5.7.3 Prospettiva metodologica: dal motion mimicry puro al controllo ibrido 67

6 Valutazione di Fattibilità Industriale, Sviluppi Futuri e Conclusioni **69**
6.1 Sintesi dei Risultati e Bilancio della Ricerca 69
6.1.1 Traguardi Raggiunti nella Pipeline Video-to-Humanoid 69
6.1.2 Dal Mocap Classico alla Visione Neurale Parametrica 70
6.1.3 Prestazioni e Tempi di Calcolo della Pipeline 70

6.2 Valutazione di Fattibilità Industriale e Impatto per Nexion Group 72
6.2.1 Flessibilità Operativa e Digitalizzazione Rapida di Tutorial Video . 72
6.2.2 La Roadmap Verso il Robot Fisico: Strategie per Colmare il Sim-to-Real Gap 73
6.2.3 Sicurezza e Coesistenza Uomo-Robot (pHRI) 73
6.3 Prospettive di Ricerca e Tecnologie Emergenti (Physical AI) 74
6.3.1 Dalla Cattura Video alla Sintesi Generativa con NVIDIA Kimodo . 74
6.3.2 Scalabilità del Whole-Body Control e Sim-to-Sim con NVIDIA ProtoMotions 76
6.3.3 Whole-Body Loco-Manipulation, Retroazione delle Forze e Agilità Sovra-Umana 77
6.3.4 Raccolta Dati su Larga Scala: Setup Egocentrici e Routine Quotidiane 78
6.3.5 Manipolazione Fine con Mani a 5 Dita e Sensori Visuotattili 79
6.3.6 La Frontiera dei Modelli Vision-Language-Action (VLA / GR00T) . 79
6.4 Considerazioni Conclusive e Trasferimento del Know-How 79
6.4.1 La Modularità Architetturale come Valore Ingegneristico 80
6.4.2 Superamento dell'Attrito Aziendale tramite la LLM-Wiki 80
6.4.3 Nexion Group e la Frontiera della Physical AI 81

A Architettura della LLM-Wiki e Knowledge Graph **83**
A.1 Architettura a Livelli, Comandi Operativi e Contraddittorio Nativo 83
A.1.1 Comandi Operativi Fondamentali 84
A.2 Governo Cognitivo: GEMINI.md, SKILL.md e Memoria con log.md 85
A.3 Estensione Applicativa: Modelli Open-Weight, Integrazione LaTeX e Modularità delle SKILL.md 86
A.4 Visualizzazione Topologica tramite Obsidian Knowledge Graph 87

Ringraziamenti **95**

Elenco delle figure

1.1 Robot umanoide compatto Unitree G1 (29 DoF) e schema dei componenti meccatronici (fonte: [32]). 4
1.2 Architettura a blocchi della pipeline software end-to-end. 5
1.3 Infrastruttura hardware impiegata per l'acquisizione video, il calcolo neurale e la simulazione dinamica. 7

2.1 Architettura logica del flusso di elaborazione di FreeMoCap. 10
2.2 Interfaccia grafica (GUI) di FreeMoCap per il controllo delle telecamere e l'elaborazione dei dati (fonte: [17]). 11
2.3 Sessioni sperimentali in officina: allestimento dell'array multi-camera, collaudo inquadrature e backstage operativo per la stazione assetto (a–g) e smontagomme (h). 13
2.4 Interfaccia di FreeMoCap per l'inserimento della dimensione metrica della tavola ChArUco (fonte: [17]). 14
2.5 Allineamento della terna globale (X, Y, Z) e del piano del pavimento tramite tavola ChArUco (fonte: [17]). 14
2.6 Sessione sperimentale di calibrazione delle telecamere con tavola ChArUco movimentata nello spazio. 15
2.7 Sessione sperimentale di calibrazione del piano del pavimento ($Z = 0$) con tavola ChArUco a terra. 15
2.8 Fasi operative del flusso di lavoro sperimentale con FreeMoCap. 15
2.9 Architettura algoritmica di triangolazione multi-camera di FreeMoCap. . . 16
2.10 Tracciamento 2D simultaneo sui quattro flussi video nell'interfaccia di FreeMoCap. 17
2.11 Allucinazione del rilevatore 2D: lo scheletro viene fittato sul sollevatore di pneumatici ignorando l'operatore reale a terra. 18
2.12 Falso positivo in assenza di persone: il modello attribuisce uno scheletro umano completo alla struttura del macchinario. 19
2.13 Ibridazione dei landmark tra l'operatore in movimento e i componenti meccanici di scena. 19

2.14 Criticità nella calibrazione del piano di terra ($Z = 0$): disallineamento della stima 3D nell'interfaccia di FreeMoCap con tre telecamere (a) e conseguente orientamento anomalo dello scheletro BVH importato in Blender (b). . . . 20
2.15 Mappatura topologica dei 33 landmark corporei di MediaPipe Pose (fonte: [8, 16]). . . . 22
2.16 Visualizzazione dell'armature BVH su Blender a seguito del collasso del tracciamento in FreeMoCap: deformazione catastrofica delle proporzioni scheletriche, alterazione delle dimensioni dei segmenti ossei e divergenza geometrica. . . . 23

3.1 Interfaccia di sistema e sessione interattiva da riga di comando per la gestione degli ambienti Conda sulla workstation. . . . 28
3.2 Diagramma architetturale dell'isolamento dei quattro ambienti Conda basati su un unico driver GPU host. . . . 30
3.3 Diagramma di flusso logico e protocolli di interscambio dati tra i tre stadi computazionali della pipeline. . . . 32

4.1 Architettura a cascata della pipeline di visione neurale monoculare GEM-X all'interno dell'ambiente isolato `env_gemx`. . . . 37
4.2 Scheletro SOMA a 77 keypoint: (a) modello teorico e (b) stima sull'operatore durante una prova in officina (fonte per (a): [28]). . . . 39
4.3 Rappresentazione scheletrica per il retargeting sui 29 gradi di libertà dell'Unitree G1 (fonte: [23]). . . . 42
4.4 Render globali di controllo generati da SOMA Retargeter: (a) scheletro umano SOMA e (b) cinematica ricalcolata per Unitree G1. . . . 42
4.5 Comando CLI per il convertitore standalone `bvh_to_csv_converter.py` con visualizzatore 3D OpenGL . . . 43
4.6 Interfaccia grafica 3D del modulo `bvhtocsv`: (a) visualizzazione affiancata per il controllo cinematico congiunto, (b) dettaglio dello scheletro SOMA estratto e (c) isolamento del moto articolare del robot Unitree G1. . . . 44
4.7 Robustezza del tracking GEM-X sotto occlusione in riprese Full HD da smartphone (a, b) e in sequenze a bassa risoluzione da FreeMoCap (c, d). . 46

5.1 Suddivisione dei livelli e ambiti operativi tra Isaac Sim (motore fisico e rendering) e Isaac Lab (framework di robot learning) (fonte: [19]). . . . 50
5.2 Rappresentazione del paradigma Sim-to-Real: addestramento massivo parallelo in simulatore GPU (Training) e successivo trasferimento della policy sul braccio robotico (fonte: [13]). . . . 51
5.3 Confronto tra i due paradigmi architetturali di NVIDIA Isaac Lab per la creazione di ambienti di addestramento (fonte: [20]). . . . 55

5.4 Esempio di Reward Hacking durante l'addestramento RL di un robot quadrupede tipo Ant: la policy massimizza la ricompensa trascinandosi su tre zampe ed evitando contatti con il quarto arto (fonte: [2]). 59
5.5 Sequenza operativa da terminale nell'ambiente `env_isaaclab`: (a) fase preliminare di addestramento intensivo e (b) successiva fase di inferenza e validazione visiva del movimento dell'Unitree G1. 61
5.6 Ambiente di simulazione 3D in Isaac Sim durante la validazione visiva (play): istanze dell'Unitree G1 che eseguono la policy di locomozione e posa su una matrice di terreni procedurali complessi. 61
5.7 Validazione qualitativa della pipeline tramite keyframe: confronto tra la posa del soggetto nel video monoculare (a) e la replica del movimento in Isaac Sim al termine del training (b). 62
5.8 Monitoraggio da terminale CLI delle tre fasi dell'addestramento PPO in Isaac Lab (traiettoria dell'operatore con utensile su pneumatico): (a) avvio iniziale, (b) consolidamento intermedio e (c) convergenza a regime. 63
5.9 Monitoraggio dell'addestramento in Isaac Lab tramite TensorBoard: (a) attivazione del server da terminale CLI e (b) interfaccia grafica per l'ispezione delle metriche di apprendimento. 65

6.1 Monitoraggio in tempo reale dell'allocazione VRAM e del carico GPU (immagine a scopo illustrativo). 71
6.2 Architettura neurale di NVIDIA Kimodo (a) e ciclo completo dal moto sintetizzato al deployment su robot umanoide reale (b) (fonte: [26]). 75
6.3 Modalità di condizionamento di NVIDIA Kimodo: dalla generazione da testo su scheletro umano SOMA (a) al controllo vincolato di traslazione radice (b), interazione con ostacoli e arredi (c), pose chiave complete (d), vincoli su organi terminali (e) e composizione sequenziale multi-prompt (f) (fonte: [26]). 76
6.4 Whole-Body Loco-Manipulation con Boston Dynamics Atlas: confronto tra esecuzione reale nel trasporto di un carico pesante (a) e addestramento per rinforzo in simulazione (b) (fonte: [3]). 77
6.5 Confronto tra prospettiva in prima persona (ego-view) e in terza persona (exo-view) nel benchmark Ego-Exo4D (fonte: [9]). 78

A.1 Architettura a tre livelli della LLM-Wiki (paradigma di Karpathy). 85
A.2 Panoramica globale del Knowledge Graph in Obsidian. 87
A.3 Dettagli del Knowledge Graph: (a) connessioni attive del nodo selezionato; (b) interconnessioni atomiche locali. 88

Elenco delle tabelle

3.1 Confronto metodologico tra ambienti virtuali Python (`venv`) ed ecosistemi Conda (`conda`). 29
3.2 Mappatura dei quattro ambienti Conda isolati costituenti l'infrastruttura software modulare. 30
3.3 Matrice sinottica di interoperabilità dell'ecosistema software modulare. . . 34

4.1 Quadro di sintesi delle prove sperimentali di validazione della pipeline GEM-X + SOMA Retargeter. 47

6.1 Resoconto prestazionale ed esecuzione dei singoli stadi della pipeline. . . . 72

Capitolo 1

Introduzione e Inquadramento nel Contesto R&D Industriale

Il presente capitolo introduce la motivazione strategica e industriale dell'adozione di robot umanoidi all'interno dei processi produttivi moderni, analizzando il superamento dell'automazione rigida a favore di sistemi antropomorfi capaci di operare in spazi di lavoro concepiti per l'essere umano. Viene inquadrato il contesto aziendale di Nexion Group e formalizzato l'obiettivo centrale della tesi: abilitare l'estrazione e il trasferimento rapido e non invasivo di abilità motorie da semplici registrazioni video 2D verso un robot umanoide a 29 gradi di libertà (Unitree G1). Vengono infine presentate la visione logica della pipeline software, la metodologia di ricerca aumentata LLM-Wiki e l'organizzazione sintetica dei successivi capitoli dell'elaborato.

1.1 Il Paradigma dell'Umanoide nei Processi Produttivi

Negli ultimi decenni, l'automazione industriale ha vissuto un'evoluzione senza precedenti, trainata dall'adozione massiva di manipolatori robotici a cinematica seriale, robot cartesiani e veicoli a guida autonoma (AGV/AMR). Tuttavia, come evidenziato nella recente rassegna trentennale sullo stato dell'arte della robotica umanoide a cura di Gu et al. [10], i paradigmi di automazione tradizionale mostrano limiti intrinseci di flessibilità quando vengono impiegati all'interno di ambienti operativi complessi, eterogenei e a rapida variabilità.

1.1.1 Limiti dell'automazione rigida e flessibilità antropomorfa

L'automazione industriale classica si fonda sul principio della strutturazione dello spazio di lavoro: per consentire a un robot tradizionale o a un cobot di eseguire un'operazione

ripetitiva con tolleranze millimetriche, l'ambiente circostante deve essere rigidamente ingegnerizzato tramite dime di posizionamento, alimentatori dedicati e barriere perimetrali di sicurezza. Qualsiasi modifica geometrica nel componente o nella sequenza di montaggio richiede costosi interventi di riconfigurazione hardware (re-layout) e lunghi fermi impianto.

La robotica umanoide introduce un cambio di paradigma radicale: invece di riadattare la fabbrica alle limitazioni del robot, si introduce un agente robotico conformato e dimensionato per operare all'interno di stazioni di lavoro antropocentriche, originariamente concepite per l'ergonomia dell'operatore umano:

- **Mobilità bipedale e navigazione multiterreno**: Capacità di superare dislivelli, gradini, pedane grigliate e corridoi stretti che precludono l'accesso a piattaforme su ruote convenzionali;
- **Manipolazione bimanuale ad ampio raggio**: Spazio di lavoro analogo a quello umano, ideale per prese coordinate, riconfigurazione per evitare auto-collisioni e movimentazione di carichi senza attrezzature speciali;
- **Intercambiabilità operativa "drop-in"**: Possibilità di operare direttamente su banchi di lavoro preesistenti e con utensili manuali standard (avvitatori, pinze, scanner), preservando la continuità operativa anche attraverso soluzioni modulari per l'autonomia energetica (quali pacchi batteria a sgancio rapido).

1.1.2 Il contesto industriale: Nexion Group e la visione dell'R&D

Il presente lavoro di tesi si inserisce all'interno delle attività di ricerca avanzata condotte presso il polo di Ricerca e Sviluppo di Nexion Group a Correggio (Reggio Emilia). Nexion rappresenta uno dei principali gruppi industriali al mondo nel settore delle attrezzature per l'assistenza e la diagnostica dei veicoli (automotive wheel service equipment), controllando marchi storici e benchmark internazionali quali Corghi (fondata nel 1960 e pioniera nell'invenzione del primo smontagomme al mondo), HPA-Faip, Mondolfo Ferro e Bright.

La genesi di questo progetto di ricerca nasce dalla visione strategica del Direttore R&D Ing. Lorenzo Vezzani e dalla consulenza specialistica dell'Ing. Bruno Ferrarini. Consapevole dell'imminente transizione verso la Physical AI e la robotica collaborativa generalista, la direzione aziendale ha promosso un'indagine esplorativa finalizzata a valutare concretamente la fattibilità, le potenzialità operative e i vincoli tecnologici legati all'introduzione di robot umanoidi nei contesti aziendali e di officina.

1.1.3 Obiettivi e perimetro del lavoro di tesi

In tale quadro, l'obiettivo cardine della tesi consiste nella progettazione, implementazione e valutazione sperimentale di una pipeline software end-to-end modulare e non invasiva, in

grado di estrarre dimostrazioni di movimento direttamente da semplici registrazioni video 2D dell'operatore (video tutorial, riprese monoculari da smartphone) e di trasferirle in modo robusto, coerente e fisicamente sostenibile sui giunti articolari di un robot umanoide bipede.

1.2 Dal Mocap Invasivo alla Visione Computazionale Non Invasiva

Il trasferimento di abilità motorie umane verso piattaforme robotiche (Human-to-Robot Motion Transfer) è stato storicamente dominato da due metodologie principali: la teleoperazione immersiva e il motion capture (mocap) ottico a marcatori.

1.2.1 Criticità operative dei sistemi mocap tradizionali in fabbrica

Sebbene accurati, i sistemi di Motion Capture ottico di grado scientifico (come Vicon o Qualisys) e le interfacce di teleoperazione immersiva presentano evidenti barriere per la scalabilità industriale:

- Impongono l'allestimento di teatri di posa dedicati con telecamere a infrarossi calibrate al millimetro;
- Richiedono che l'operatore indossi tute aderenti con marker riflettenti o esoscheletri ingombranti, alterando la naturalezza del gesto e rallentando i cicli di lavoro;
- Comportano costi hardware elevati e complessità di calibrazione incompatibili con la dinamicità dei reparti produttivi.

1.2.2 L'opportunità dei modelli di visione per il trasferimento del moto

La sfida affrontata in questo elaborato è la transizione verso un paradigma completamente markerless e non invasivo: sfruttare i moderni modelli di Visione Artificiale e Deep Learning per trasformare archivi di video 2D standard in traiettorie cinematiche tridimensionali direttamente utilizzabili per il controllo robotico. Questo approccio consente di digitalizzare in modo trasparente il know-how manuale degli operatori esperti senza interrompere i flussi di lavoro e senza richiedere sensoristica indossabile specialistica.

1.2.3 La piattaforma target: Unitree G1

Come piattaforma di riferimento hardware e simulativa per l'intera pipeline è stato selezionato il robot umanoide compatto Unitree G1 [32], illustrato in Figura 1.1.

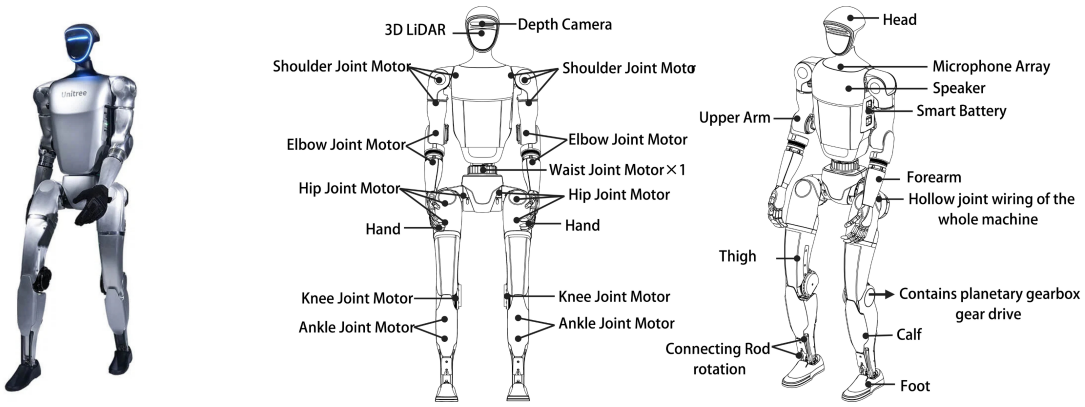


Figura 1.1: Robot umanoide compatto Unitree G1 (29 DoF) e schema dei componenti meccatronici (fonte: [32]).

Con un'altezza di circa 1.27 m, una massa di circa 35 kg e un'articolazione a 29 gradi di libertà (6 DoF per gamba, 7 DoF per braccio, 3 DoF per torso/vita), il G1 offre un'eccellente combinazione tra compattezza, sicurezza nelle interazioni fisiche uomo-robot (pHRI) e supporto nativo a formati standard aperti (URDF e OpenUSD) per la simulazione fisica su GPU.

1.3 Panoramica della Pipeline Proposta e Metodologia di Ricerca

1.3.1 Schema logico della pipeline end-to-end

L'infrastruttura tecnologica sviluppata nel corso della tesi si articola nella catena modulare e disaccoppiata illustrata in Figura 1.2:

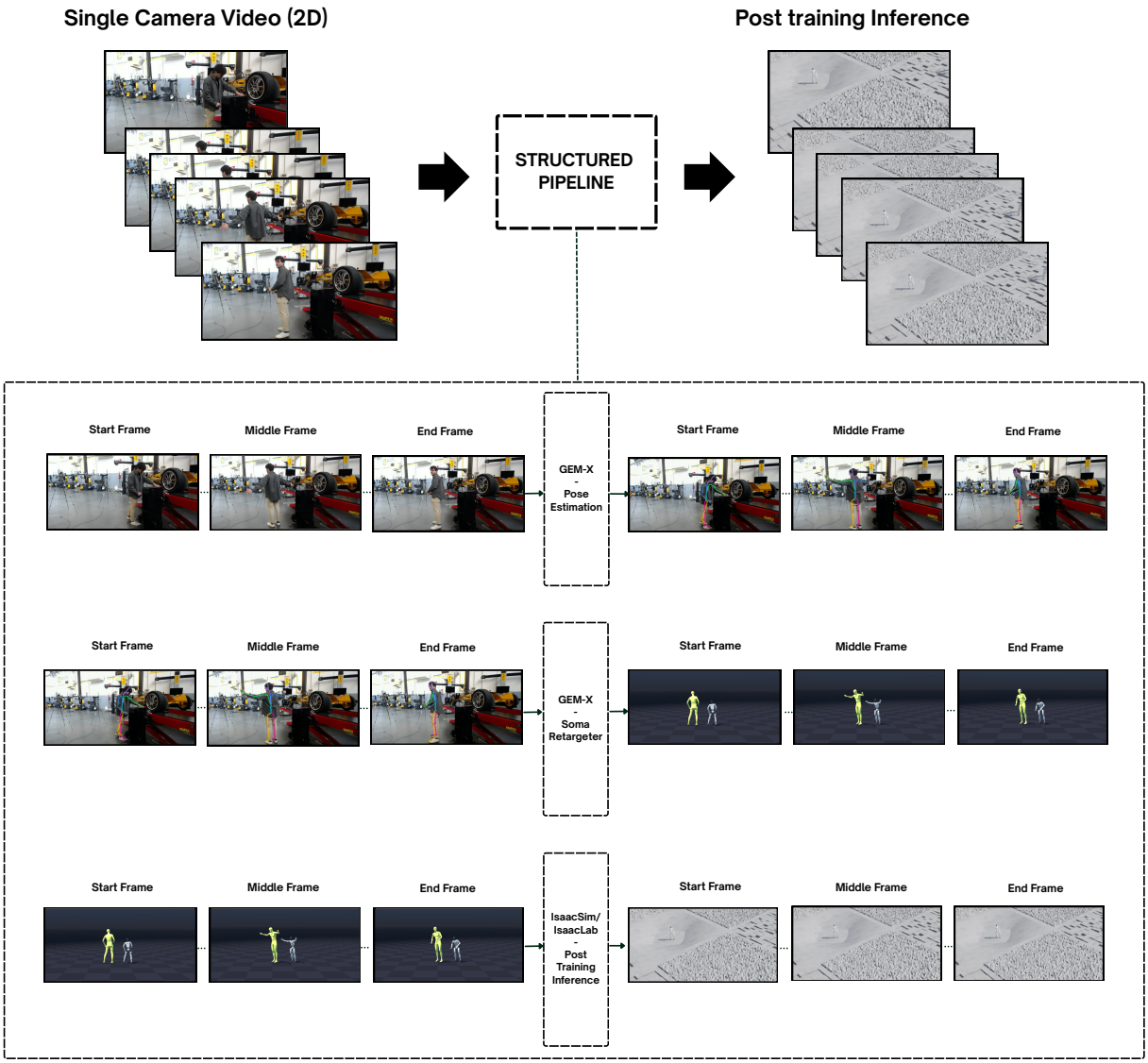


Figura 1.2: Architettura a blocchi della pipeline software end-to-end.

Il flusso operativo procede attraverso quattro stadi sequenziali e modulari:

1. **Acquisizione Video 2D**: Registrazione RGB standard dell'operatore umano durante l'esecuzione del compito da replicare;

2. **Stima Neurale della Posa 3D (NVIDIA GEM-X)**: Ricostruzione della cinematica tridimensionale del corpo umano a partire dal flusso video monoculare;

3. **Kinematic Retargeting (SOMA Retargeter)**: Adattamento geometrico e articolare delle traiettorie umane sulla struttura cinematica del robot umanoide;

4. **Addestramento e Validazione in Simulazione Dinamica (NVIDIA Isaac Lab)**: Addestramento delle policy di controllo motorio (Deep Reinforcement Learning), verifica del comportamento fisico del robot e validazione delle prestazioni in ambiente simulato.

1.3.2 Metodologia di ricerca aumentata e trasferimento del know-how (LLM-Wiki)

Data la rapidità con cui evolvono i modelli di visione, diffusione del moto (es. NVIDIA Kimodo [26]) e controllo neurale per la Physical AI, l'intero processo di indagine scientifica è stato condotto adottando una metodologia di gestione della conoscenza aumentata da Large Language Model, strutturata secondo il paradigma LLM-Wiki (Karpathy Architecture).

Oltre a garantire la tracciabilità rigorosa e l'integrità dei riferimenti bibliografici, la Wiki costituisce un elaborato tangibile di trasferimento tecnologico (Corporate Knowledge Transfer) per il reparto R&D di Nexion Group, rendendo la base di conoscenza persistente, interrogabile e continuamente estendibile per le successive iterazioni del progetto. La trattazione dettagliata dell'architettura e delle metriche della Wiki è demandata all'Appendice A.

1.3.3 Infrastruttura hardware di calcolo e apparati di acquisizione

L'implementazione, l'addestramento e la validazione di una pipeline complessa di Physical AI richiedono un'adeguata infrastruttura di calcolo in grado di sostenere carichi paralleli intensivi sia sul fronte dell'inferenza neurale che della simulazione fisica dinamica:

- **Workstation mobile ad alte prestazioni (ASUS ROG Strix SCAR 18)**: Come piattaforma principale di sviluppo e calcolo è stato impiegato un laptop ad alte prestazioni equipaggiato con processore Intel Core Ultra 9 290HX Plus, scheda grafica dedicata NVIDIA GeForce RTX 5080 Laptop GPU, display da 18" HDR 4K, 32 GB di memoria RAM DDR5 e unità a stato solido NVMe PCIe da 1 TB (Figura 1.3a). La presenza dei Tensor Core e dei Ray Tracing Core di ultima generazione dell'architettura NVIDIA garantisce la capacità di calcolo e la larghezza di banda di memoria VRAM necessarie per eseguire agevolmente ambienti computazionalmente onerosi quali NVIDIA Isaac Sim e Isaac Lab [18], oltre a supportare la compilazione JIT di kernel differenziabili in NVIDIA Warp [21] e l'inferenza di Vision Transformers pesanti (GEM-X [14] e ViTPose [33]);

- **Dispositivo di acquisizione ottica monoculare (Webcam Logitech C920)**: Per la registrazione dei filmati video 2D dell'operatore in ambiente reale è stata adottata una webcam commerciale Logitech C920 HD Pro (Figura 1.3b), con sensore ottico Full HD (1080p a 30 fps) e ottica grandangolare con messa a fuoco automatica. L'impiego di una periferica di fascia consumer a basso costo risponde all'obiettivo strategico di dimostrare la flessibilità e la scalabilità della pipeline proposta, attestando la possibilità di estrarre dimostrazioni cinematiche accurate da normali

flussi video senza la necessità di investire in costosi sistemi di ripresa industriali o telecamere calibrate di laboratorio.



(a) Workstation mobile ASUS ROG Strix SCAR 18 (RTX 5080, Intel Core Ultra 9).



(b) Webcam commerciale Logitech C920 HD Pro per acquisizione video 2D.

Figura 1.3: Infrastruttura hardware impiegata per l'acquisizione video, il calcolo neurale e la simulazione dinamica.

1.4 Struttura dell'Elaborato

Il presente documento è organizzato nei seguenti capitoli:

- **Capitolo 2 – Studio Sperimentale su FreeMoCap: Setup, Test e Analisi dei Colli di Bottiglia**: Documenta la fase sperimentale preliminare condotta con sistema markerless multi-camera open-source, analizzandone la complessità di calibrazione, i colli di bottiglia computazionali e le cause dell'incompatibilità topologica del file BVH per il controllo robotico;
- **Capitolo 3 – Architettura Software, Gestione delle Dipendenze ed Ecosistema Modulare**: Descrive la progettazione dell'infrastruttura a micro-ambienti Conda isolati e la formalizzazione dei protocolli di interscambio dati standardizzati per garantire la massima riproducibilità in ambiente R&D;
- **Capitolo 4 – Visione Monoculare Neurale e Kinematic Retargeting con GEM-X e SOMA**: Delinea l'implementazione del modello di stima monoculare 3D e dell'ottimizzatore cinematico su GPU basato su vincoli di contatto e limiti di giunto per Unitree G1;

- **Capitolo 5 – Proof of Concept in Simulazione Dinamica (Isaac Lab) e Analisi Critica dell'Imitazione**: Formalizza l'ambiente di simulazione fisica GPU, l'impostazione del task di tracking articolare con Deep Reinforcement Learning e la validazione empirica su terreni complessi, analizzando criticamente i limiti dell'imitazione puramente antropomorfa rispetto all'agilità meccanica del robot;
- **Capitolo 6 – Valutazione di Fattibilità Industriale, Sviluppi Futuri e Conclusioni**: Presenta la sintesi quantitativa e qualitativa dei risultati, i benchmark computazionali di filiera, l'analisi di fattibilità e sicurezza pHRI per Nexion Group, le prospettive di frontiera (modelli generativi NVIDIA Kimodo, whole-body control ProtoMotions [30], raccolta dati egocentrica con telecamere frontali) e il ruolo strategico della LLM-Wiki nel trasferimento permanente del know-how.

Capitolo 2

Studio Sperimentale su FreeMoCap: Setup, Test e Analisi dei Colli di Bottiglia

Il presente capitolo documenta la prima fase sperimentale della ricerca, focalizzata sulla valutazione operativa di FreeMoCap come sistema di motion capture markerless multi-camera. Vengono descritti l'allestimento dell'ambiente software dedicato (env_freemocap), la procedura di calibrazione ottica mediante scacchiera di calibrazione geometrica e le prove condotte con array di tre e quattro telecamere su movimenti di prova e task operativi reali. L'analisi si concentra sui colli di bottiglia computazionali, sulla vulnerabilità a falsi positivi ambientali e, in particolare, sulle incompatibilità topologiche e cinematiche che precludono l'utilizzo diretto dei file BVH generati all'interno dei software di retargeting per robot umanoidi. Il capitolo formalizza il valore metodologico di questo risultato negativo, identificando le ragioni scientifiche del cambio di paradigma verso modelli neurali parametrici su GPU.

2.1 Architettura e Setup del Banco di Prova Multi-Camera

Al fine di valutare la fattibilità dell'acquisizione del movimento umano mediante strumenti open-source a basso costo, la prima fase sperimentale del progetto si è concentrata sull'adozione e l'integrazione di FreeMoCap [17]. L'architettura del software si basa su una pipeline ibrida che combina modelli di Deep Learning per la visione 2D monoculare e algoritmi di geometria epipolare classica per la ricostruzione 3D multi-vista.

Come illustrato schematicamente nel diagramma di flusso logico di Figura 2.1, il processamento dell'immagine si articola nei seguenti stadi sequenziali:

1. **Segmentazione e Bounding Box (Rete Neurale YOLOv8)**: Il frame video grezzo $I \in \mathbb{R}^{H \times W \times 3}$ (dove H indica l'altezza in pixel, W la larghezza e 3 i canali cromatici RGB) di ciascuna telecamera viene processato da una rete neurale profonda di rilevamento oggetti (YOLOv8). Il suo compito è localizzare la persona ed estrarre un ritaglio (crop) delimitato dal bounding box (x, y, w, h) (dove (x, y) sono le coordinate pixel dell'angolo superiore sinistro, w la larghezza e h l'altezza della regione rettangolare), isolando l'operatore dallo sfondo circostante;

2. **Stima di Posa 2D (Rete Neurale MediaPipe Pose)**: L'immagine ritagliata viene inviata a una seconda rete neurale profonda convoluzionale (MediaPipe Pose). La rete esegue la regressione spaziale delle coordinate pixel (u_j, v_j) (dove u_j è la coordinata orizzontale e v_j la coordinata verticale nel piano immagine) di 33 punti chiave anatomici (landmark corporei superficiali) associati a ciascun frame monoculare;

3. **Triangolazione Multi-Vista (Solutore Geometrico DLT)**: A questo livello la pipeline non impiega ulteriori reti neurali, bensì un solutore geometrico deterministico basato sulla Triangolazione Lineare Diretta (Direct Linear Transformation, DLT) con pesatura esponenziale. Il modulo fonde i raggi ottici convergenti delle N telecamere calibrate per ricavare la posizione cartesiana tridimensionale $p_j = (X_j, Y_j, Z_j)$ di ciascun landmark nello spazio globale;

4. **Ricostruzione Scheletrica ed Esportazione (.BVH / .CSV)**: Le coordinate tridimensionali vengono infine organizzate secondo una gerarchia di segmenti ossei e rotazioni locali dei giunti, esportando le serie temporali cinematiche nei formati standard `.BVH` (Biovision Hierarchy) e `.CSV`.

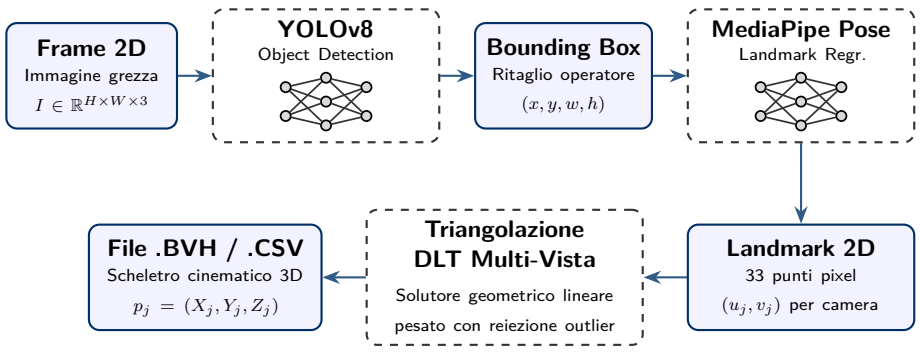


Figura 2.1: Architettura logica del flusso di elaborazione di FreeMoCap.

2.1.1 Configurazione dell'ambiente software dedicato e dell'array di telecamere

Per consentire l'esecuzione stabile della suite FreeMoCap e del suo stack grafico e algoritmico, è stato predisposto sulla workstation di sviluppo un ambiente virtuale Conda isolato denominato *env_freemocap*. Tale ambiente raggruppa un interprete Python 3.9/3.10, i framework di visione OpenCV e MediaPipe per la stima della posa, nonché l'interfaccia grafica interattiva (mostrata in Figura 2.2) impiegata per il rilevamento delle telecamere, il monitoraggio delle registrazioni e l'avvio della pipeline di elaborazione. La trattazione approfondita delle specifiche di configurazione di tale ambiente, unitamente alla risoluzione delle dipendenze software e alla sua collocazione all'interno dell'architettura multi-ambiente complessiva, sarà formalizzata organicamente nel Capitolo 3.

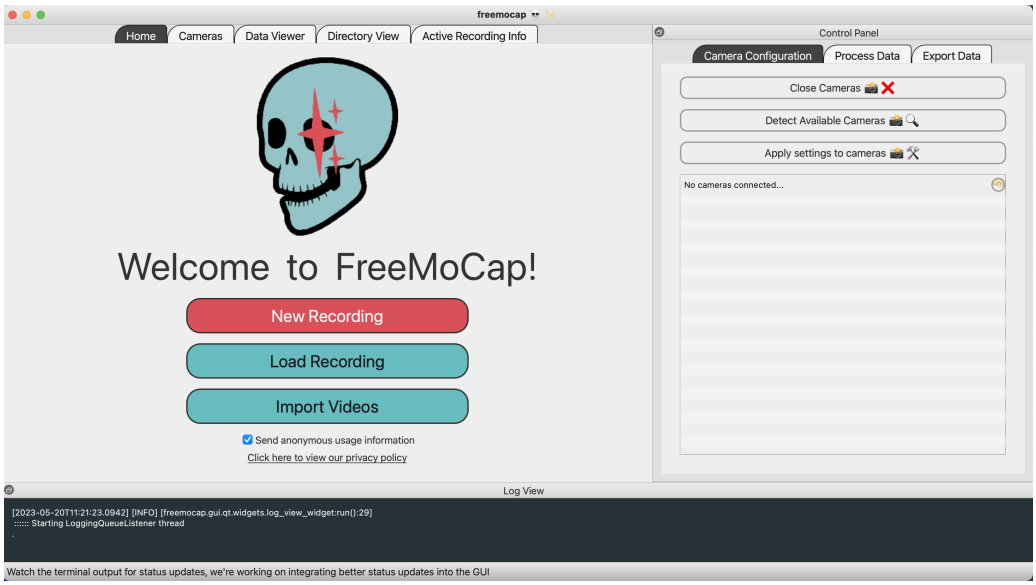


Figura 2.2: Interfaccia grafica (GUI) di FreeMoCap per il controllo delle telecamere e l'elaborazione dei dati (fonte: [17]).

Il banco di prova sperimentale è stato allestito direttamente in ambiente industriale reale all'interno di un'officina meccanica specializzata, predisponendo un array di telecamere commerciali (webcam ad alta definizione) montate su stativi regolabili e impostate alla risoluzione standard di 640×480 pixel a 30 fps. La disposizione ottica è stata dimensionata per coprire interamente il volume di lavoro dell'operatore, garantendo un'adeguata ridondanza e molteplici prospettive angolari. La sperimentazione ha previsto l'esplorazione di due configurazioni ottiche:

- **Array a tre telecamere**: Disposizione triangolare attorno alla postazione con angoli di incidenza reciproci di circa 120°, concepita per minimizzare l'ingombro hardware pur garantendo la ridondanza minima per la triangolazione spaziale;

- **Array a quattro telecamere**: Disposizione quadrangolare ai quattro vertici dello spazio di cattura, finalizzata a mitigare le zone d'ombra e le auto-occlusioni degli arti superiori durante i compiti di manipolazione.

Come documentato nella raccolta fotografica di Figura 2.3, la campagna sperimentale sul campo si è articolata attraverso le seguenti fasi operative:

1. **Allestimento e Sincronizzazione Hardware (Figure 2.3a–2.3c)**: Posizionamento dei treppiedi a terra e sopraelevati direttamente sulle pedane del ponte sollevatore della stazione di assetto ruote, al fine di garantire angoli di vista convergenti sia a quota pavimento che laterali. Il software FreeMoCap si occupa direttamente di gestire l'avvio simultaneo della registrazione e la sincronizzazione temporale dei flussi video tra tutte le telecamere collegate;

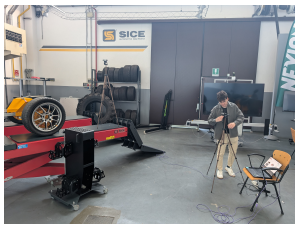
2. **Collaudo Inquadrature e Volume di Cattura (Figure 2.3d–2.3e)**: Sessioni preliminari di test condotte dall'autore per verificare che l'intero raggio di mobilità dell'operatore (inclusa la massima estensione degli arti superiori e i movimenti di flessione del busto) rimanesse pienamente inscritto nel campo visivo condiviso da tutte le telecamere;

3. **Sessioni Operative Reali e Backstage (Figure 2.3f–2.3g)**: Ripresa di sessioni di lavoro reali con un operatore industriale esperto impegnato nella manovra del carrello di sollevamento pneumatici, nel posizionamento della ruota e nelle regolazioni fini sulla vettura posizionata sul ponte d'assetto;

4. **Estensione del Setup ad Altre Postazioni Operative (Figura 2.3h)**: Per validare la versatilità dell'infrastruttura di ripresa su compiti eterogenei di mobilitazione e servizio ruote, l'array multi-camera è stato allestito anche presso la stazione dello smontagomme industriale, posizionando la tavola ChArUco sul pavimento per la calibrazione metrica dell'area di lavoro.



(a) Allestimento treppiedi assetto.



(b) Postazione di controllo.



(c) Telecamera su ponte.



(d) Collaudo inquadratura.



(e) Verifica volume cattura.



(f) Manovra carrello ruota.



(g) Sessione con operatore.



(h) Setup smontagomme.

Figura 2.3: Sessioni sperimentali in officina: allestimento dell'array multi-camera, collaudo inquadrature e backstage operativo per la stazione assetto (a–g) e smontagomme (h).

2.1.2 Calibrazione intrinseca/estrinseca con tavola ChArUco e piano del pavimento

La ricostruzione geometrica multi-vista richiede una rigorosa fase di calibrazione preliminare. Prima dell'ingresso in scena dell'operatore, è stata eseguita la procedura di taratura ottica utilizzando una tavola ChArUco 5×3 (composta da una scacchiera alternata a marcatori ArUco unici):

1. **Misurazione metrica di punti geometrici noti**: All'interno dell'interfaccia di configurazione di FreeMoCap (illustrata in Figura 2.4) è necessario inserire la dimensione metrica reale misurata (in millimetri) del lato dei quadrati (square size), consentendo all'algoritmo di stabilire il fattore di scala reale per la ricostruzione tridimensionale;

2. **Calibrazione dei parametri intrinseci ed estrinseci**: Movimentando la tavola ChArUco all'interno del volume condiviso dalle telecamere (Figura 2.6), il solutore individua i marker ArUco e i vertici della scacchiera, stimando le distanze focali, i centri ottici, i coefficienti di distorsione lenticolare e le matrici di rototraslazione reciproche tra i punti di vista;

3. **Allineamento del sistema di coordinate e piano del pavimento ($Z = 0$)**: Posizionando rigidamente la tavola a terra con un orientamento prestabilito sul pavimento (secondo lo schema teorico in Figura 2.5 e la procedura sperimentale in Figura 2.7), il software individua il marker di origine (0), allinea gli assi X e Y

lungo i bordi della tavola e orienta l'asse verticale Z lungo il vettore normale uscente, vincolando la quota del pavimento a $Z = 0$.

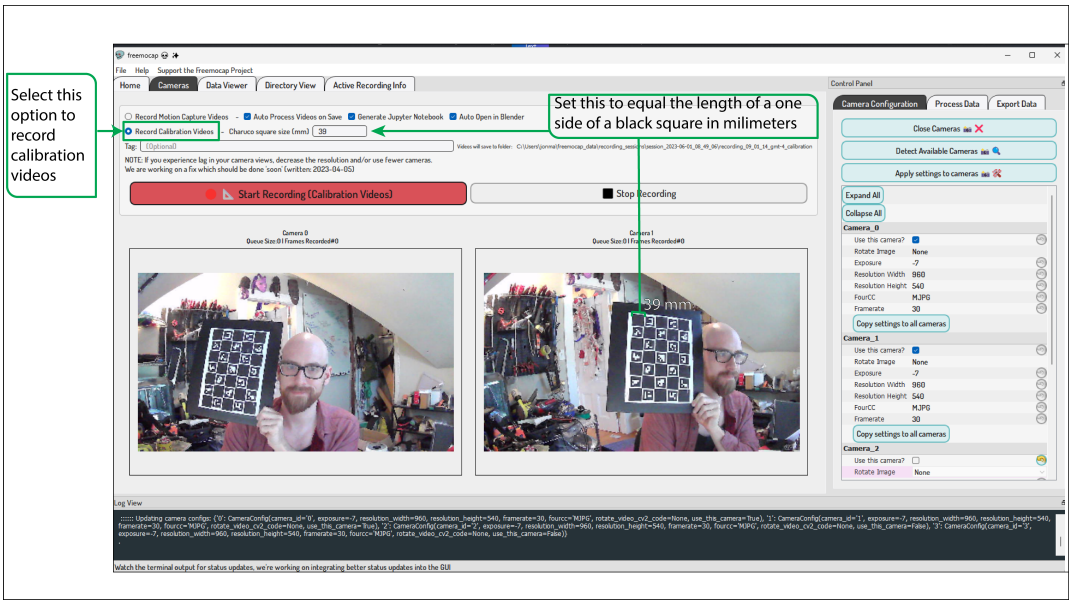


Figura 2.4: Interfaccia di FreeMoCap per l'inserimento della dimensione metrica della tavola ChArUco (fonte: [17]).

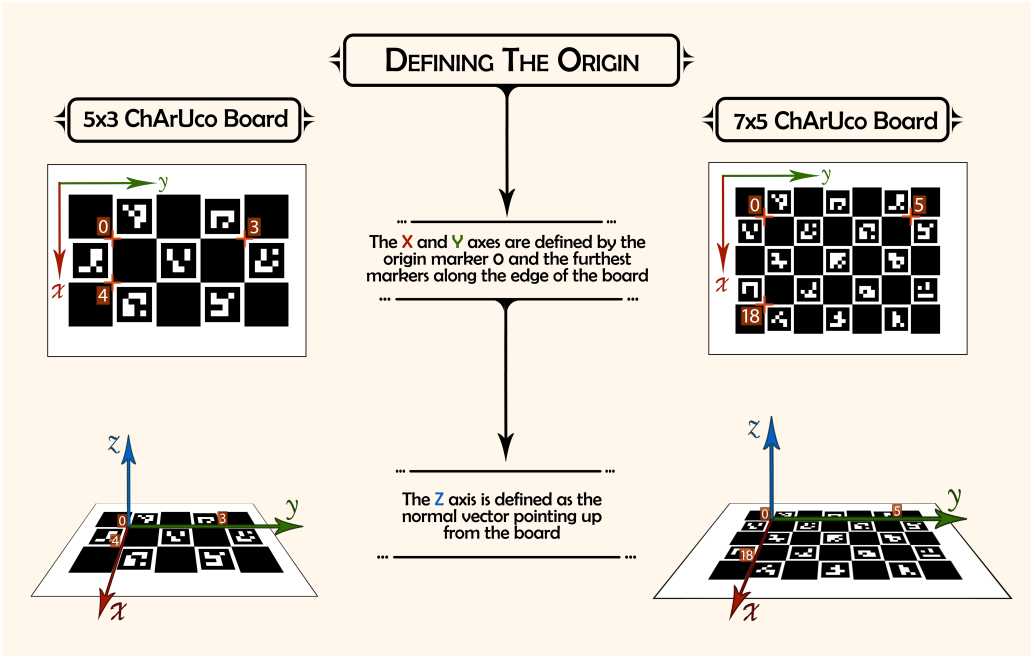
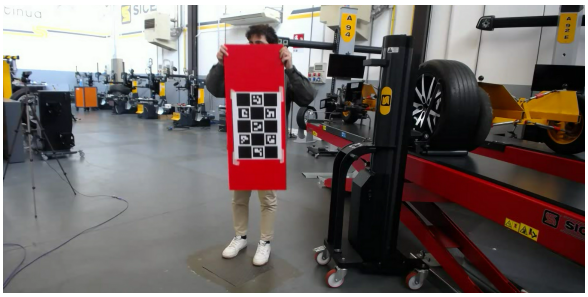
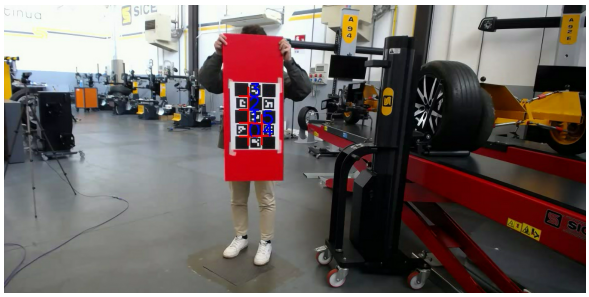


Figura 2.5: Allineamento della terna globale (X, Y, Z) e del piano del pavimento tramite tavola ChArUco (fonte: [17]).

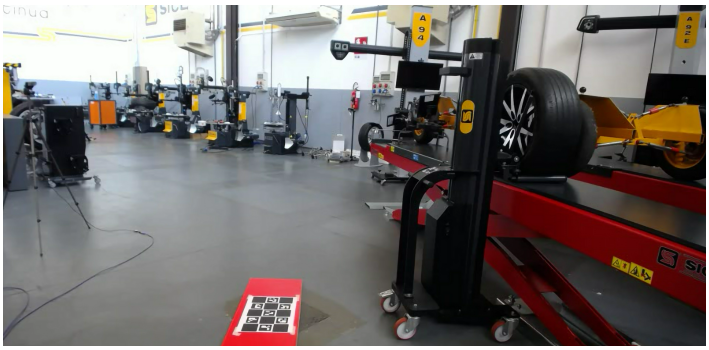


(a) Movimentazione della tavola nel volume.



(b) Rilevamento e annotazione dei vertici.

Figura 2.6: Sessione sperimentale di calibrazione delle telecamere con tavola ChArUco movimentata nello spazio.



(a) Posizionamento della tavola sul pavimento.



(b) Identificazione e annotazione del piano.

Figura 2.7: Sessione sperimentale di calibrazione del piano del pavimento ($Z = 0$) con tavola ChArUco a terra.

Le fasi operative complessive del lavoro sperimentale condotto con FreeMoCap sono sintetizzate nel diagramma di flusso illustrato in Figura 2.8.



Figura 2.8: Fasi operative del flusso di lavoro sperimentale con FreeMoCap.

2.1.3 Pipeline algoritmica di triangolazione DLT e reiezione degli outlier

Una volta sincronizzati i flussi video registrati, la catena di elaborazione di FreeMoCap (schematizzata nel diagramma concettuale di Figura 2.9) esegue l'inferenza multi-camera operando in parallelo sui canali video (Figura 2.10) attraverso i seguenti stadi algoritmici:

1. **Rilevamento e ritaglio 2D**: Un modello YOLOv8 individua la regione di interesse (bounding box) dell'operatore in ogni singolo frame;

2. **Stima della posa 2D**: All'interno del ritaglio, MediaPipe Pose estrae le coordinate pixel (u_j, v_j) di 33 landmark corporei bidimensionali per ciascuna telecamera;

3. **Triangolazione Lineare Diretta (DLT) con reiezione outlier**: Le proiezioni bidimensionali convergenti vengono trasformate in coordinate tridimensionali cartesiane $p_j = (X_j, Y_j, Z_j)$ nello spazio globale. In questo contesto, con il termine outlier si identifica una stima 2D fortemente errata generata da una telecamera (ad esempio a causa di occlusioni o riflessi che deviano il landmark su elementi di sfondo). Per evitare che una misura anomala distorca l'intera media spaziale, FreeMoCap calcola l'errore di riproiezione ε_i di ciascuna camera e applica una pesatura esponenziale:

$$w_i = \exp\left(-5 \cdot \frac{\varepsilon_i}{\varepsilon_{\text{target}}}\right), \quad \hat{p}_j = \frac{\sum_{i=1}^{N_{\text{cam}}} w_i \, p_{i,j}}{\sum_{i=1}^{N_{\text{cam}}} w_i} \tag{2.1}$$

   In tal modo, le telecamere affette da forte errore ricevono un peso quasi nullo ($w_i \approx 0$), escludendo di fatto il dato aberrante dal calcolo della coordinata tridimensionale finale $\hat{p}_j$;

4. **Esportazione dei file di moto**: La traiettoria dei 33 landmark viene infine salvata in formato tabellare (CSV) e convertita in uno scheletro gerarchico standard (.BVH).

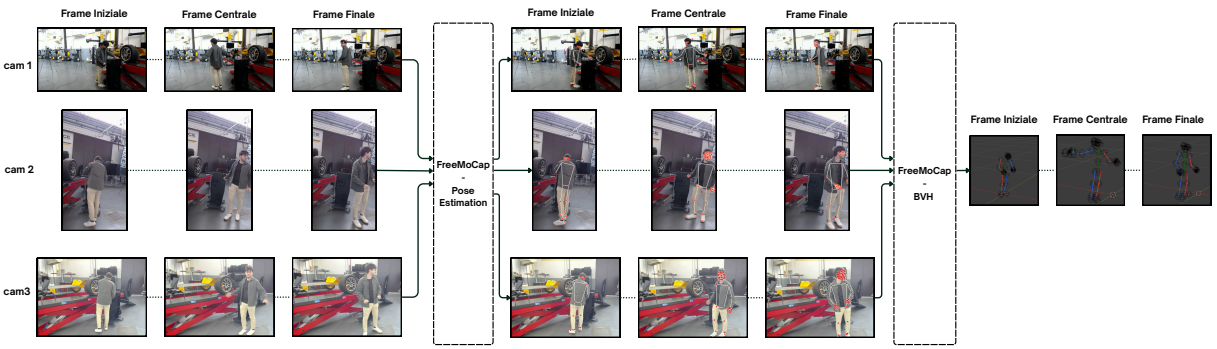


Figura 2.9: Architettura algoritmica di triangolazione multi-camera di FreeMoCap.

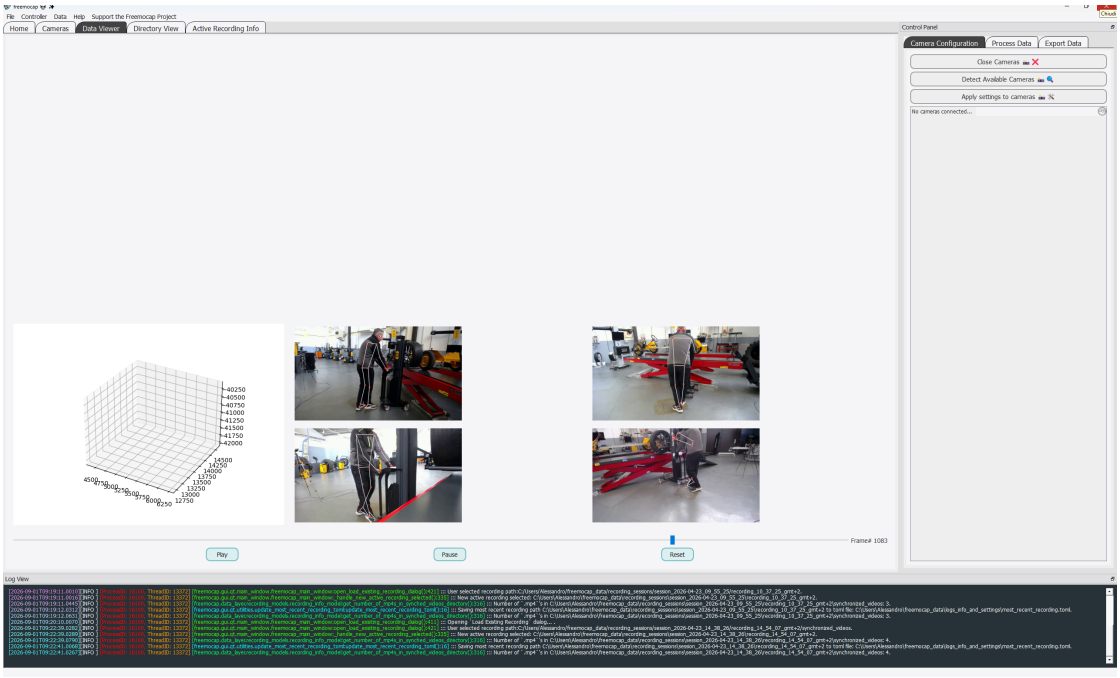


Figura 2.10: Tracciamento 2D simultaneo sui quattro flussi video nell'interfaccia di FreeMoCap.

2.2 Risultati Sperimentali e Limiti Operativi in Ambiente Reale

Le prove sperimentali condotte sul campo hanno messo in luce severe limitazioni di robustezza ottica e scalabilità computazionale, che ne compromettono l'efficacia all'interno di scenari industriali e di laboratorio.

2.2.1 Sensibilità alle condizioni ambientali, falsi positivi e perdita di tracciamento

L'ambiente operativo reale di officina ha messo in luce la vulnerabilità della catena di rilevamento 2D (bounding box YOLO e landmarking MediaPipe) durante l'elaborazione parallela dei flussi (Figura 2.10), manifestando sistematiche allucinazioni visive dovute alla complessità geometrica dei macchinari circostanti:

- **Sostituzione del soggetto con macchinari di scena**: Quando l'operatore si muoveva all'interno dello spazio di lavoro (ad esempio per posizionare la scacchiera a terra, come illustrato in Figura 2.11), il modello di bounding box allucinava la presenza umana su un sollevatore di pneumatici verticale, forzando la rete di posa a ricostruire un intero scheletro anatomico sui componenti meccanici e ignorando completamente il soggetto reale;

- **Allucinazione della posa in assenza di persone**: Il medesimo errore si manifestava con regolarità anche a scena completamente vuota (Figura 2.12), dove la macchina continuava a essere classificata come un operatore eretto, confermando la necessità di ambienti iper-strutturati (sfondi neutri, assenza di attrezzi e abbigliamento a contrasto marcato);

- **Frammentazione e ibridazione dei landmark**: Durante l'esecuzione dei compiti operativi, il modello tendeva a "spalmare" i 33 punti corporei tra l'uomo e gli oggetti (Figura 2.13), ancorando ad esempio testa e spalle al cerchione del macchinario e le gambe all'operatore;

- **Corruzione catastrofica della triangolazione 3D**: Sebbene la pesatura esponenziale dell'Equazione (2.1) riesca a filtrare l'errore isolato di una singola telecamera, nei contesti di officina più punti di vista allucinavano contemporaneamente sul medesimo macchinario. Gli outlier diventavano così maggioritari, mandando in crisi il solutore e producendo traiettorie 3D completamente distorte e non fisiche.

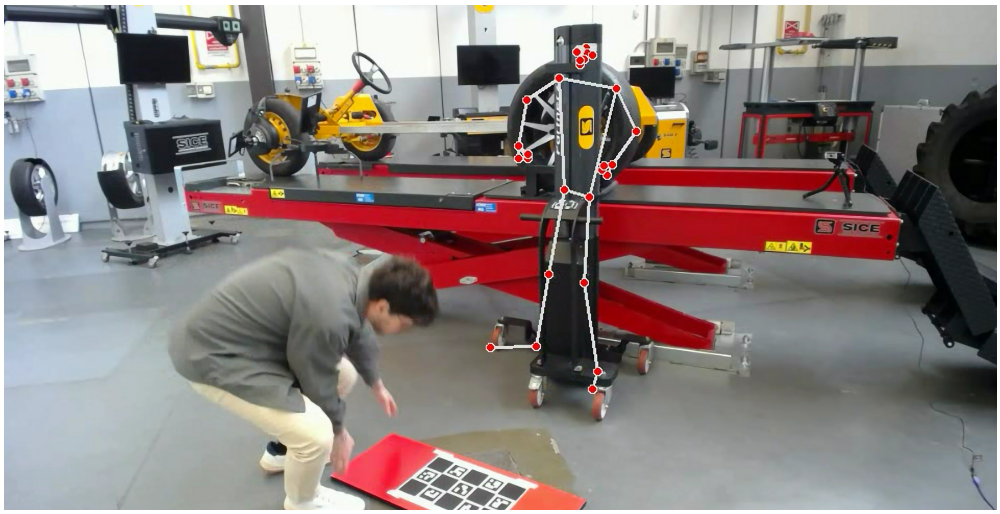


Figura 2.11: Allucinazione del rilevatore 2D: lo scheletro viene fittato sul sollevatore di pneumatici ignorando l'operatore reale a terra.



Figura 2.12: Falso positivo in assenza di persone: il modello attribuisce uno scheletro umano completo alla struttura del macchinario.

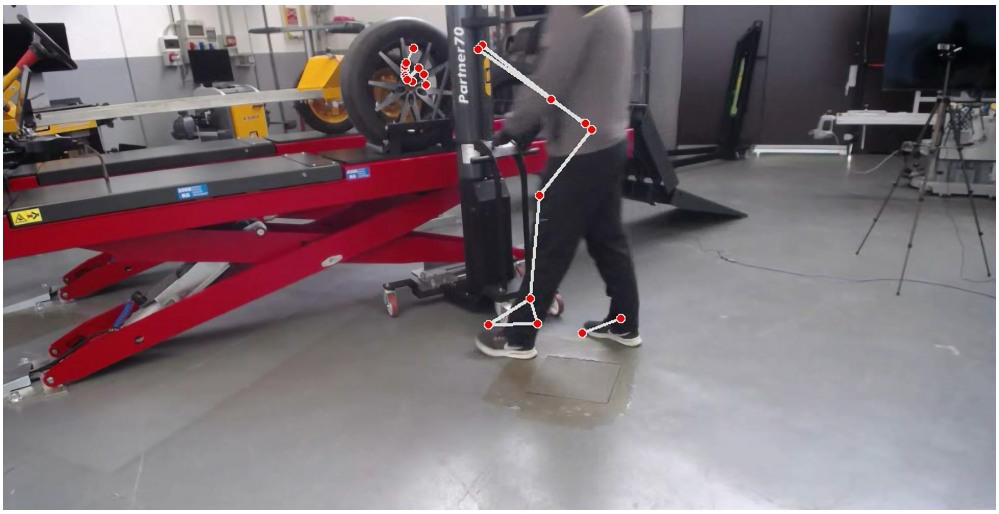


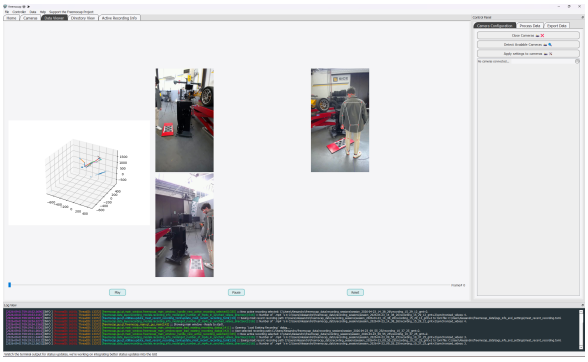
Figura 2.13: Ibridazione dei landmark tra l'operatore in movimento e i componenti meccanici di scena.

2.2.2 Instabilità nella definizione del piano del suolo ($Z = 0$)

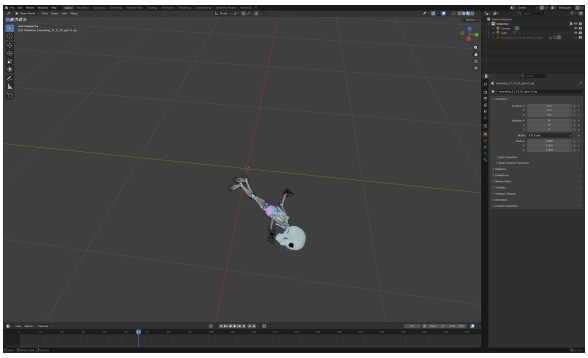
Nonostante la calibrazione con la scacchiera sia stata eseguita seguendo scrupolosamente le linee guida ufficiali, il sistema ha manifestato significative criticità nella corretta collocazione del piano di terra, come evidenziato nel confronto sperimentale di Figura 2.14:

- **Disallineamento geometrico in FreeMoCap**: Come visualizzato nell'interfaccia di FreeMoCap (Figura 2.14a), la stima dei raggi ottici dalle tre telecamere produce una terna globale ruotata e traslata in modo errato. Nel grafico cartesiano 3D (riquadro sinistro) il soggetto non risulta vincolato stabilmente alla quota $Z = 0$, manifestando uno sprofondamento parziale dei landmark podalici e continue oscillazioni verticali (foot-skating);

- **Propagazione dell'errore sul file BVH in Blender**: Tale incoerenza geometrica si riflette direttamente sulla gerarchia cinematica esportata. Aprendo il file `.BVH` nell'ambiente di modellazione Blender (Figura 2.14b), l'armature scheletrica eredita l'orientamento obliquo e l'interpenetrazione con la griglia del pavimento;
- **Incompatibilità con il controllo robotico**: In un'ottica di trasferimento robotico su un umanoide bipede (quale Unitree G1), dove la dinamica del contatto al suolo e la stabilizzazione del baricentro sono parametri critici, l'assenza di un piano di appoggio rigidamente consistente rende il dato cinematico inutilizzabile per il controllo dinamico.



(a) Calibrazione errata del piano in FreeMoCap (soggetto mal vincolato a $Z = 0$).



(b) Scheletro BVH su Blender con orientamento errato rispetto al suolo.

Figura 2.14: Criticità nella calibrazione del piano di terra ($Z = 0$): disallineamento della stima 3D nell'interfaccia di FreeMoCap con tre telecamere (a) e conseguente orientamento anomalo dello scheletro BVH importato in Blender (b).

2.2.3 Collo di bottiglia computazionale su CPU e instabilità di sistema

L'architettura software di FreeMoCap si affida a moduli di multiprocessing su CPU per la decodifica, il ritaglio e l'inferenza dei singoli flussi video. Nelle prove pratiche condotte sul campo, questo paradigma ha evidenziato un severo collo di bottiglia prestazionale:

- **Vincolo di risoluzione forzata a 640×480 pixel**: Sebbene le webcam commerciali impiegate supportassero nativamente la risoluzione Full HD (1920×1080 a 30 fps), tutti i filmati sono stati registrati e processati alla risoluzione ridotta di 640×480 pixel. Tentativi preliminari di elevare la qualità video al formato 1080p con più telecamere attive in parallelo hanno provocato la saturazione immediata della memoria RAM, il blocco del gestore dei processi Python e il conseguente crash completo dell'applicativo durante la fase di estrazione dei landmark di MediaPipe;
- **Compromesso tra stabilità computazionale e dettaglio ottico**: La necessità di degradare la risoluzione a 640×480 per consentire il completamento della

pipeline ha comportato una drastica perdita di definizione sui dettagli fini (quali l'orientamento delle dita, i punti di contatto del piede e le articolazioni distali), peggiorando ulteriormente la precisione della stima 2D e la robustezza geometrica della triangolazione;

- **Tempi di calcolo offline proibitivi**: Anche alla risoluzione minimale di 640×480, l'elaborazione di sequenze di movimento di appena 10–15 secondi ha richiesto tempi di calcolo offline compresi tra i 20 e i 35 minuti sulla CPU della workstation. Passando dall'array a tre all'array a quattro telecamere, l'overhead computazionale è cresciuto in modo non lineare, confermando l'inadeguatezza di un'architettura CPU-bound per l'elaborazione continuativa di dataset industriali.

2.3 L'Ostacolo Critico: Incompatibilità Cinematica e la Sfida del Retargeting

Al di là delle problematiche di calibrazione e di sovraccarico computazionale, l'ostacolo più profondo emerso dalla sperimentazione riguarda la natura stessa dei dati generati da FreeMoCap e la loro intrinseca incompatibilità con i requisiti del kinematic retargeting.

2.3.1 Mismatch topologico tra landmark cartesiani e requisiti dei motori di retargeting

Il retargeting cinematico rappresenta il processo computazionale incaricato di mappare le traiettorie di moto di un soggetto sorgente (il corpo umano) sulla struttura articolata di un robot target, quale l'Unitree G1 a 29 gradi di libertà [32].

I moderni motori e solutori di retargeting richiedono in ingresso gerarchie articolari rigorosamente strutturate, in cui siano definiti i centri di rotazione fisici dei giunti e le relative matrici di rotazione nello spazio tridimensionale $SO(3)$. Al contrario, la pipeline di FreeMoCap produce una collezione di 33 landmark cartesiani sparsi e superficiali (illustrati in Figura 2.15), privi di informazioni di orientamento locale.

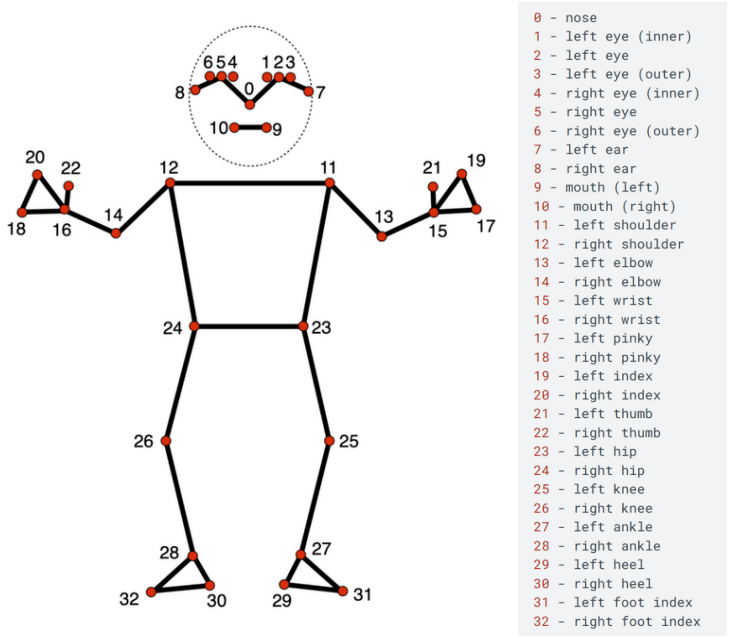


Figura 2.15: Mappatura topologica dei 33 landmark corporei di MediaPipe Pose (fonte: [8, 16]).

Questo divario topologico genera severe complicazioni metodologiche:

1. **Mismatch di interfaccia con i software di retargeting**: I comuni software e framework di retargeting per la robotica si aspettano modelli di scheletro standardizzati con giunzioni rotoidali conformi. I punti derivati da MediaPipe non coincidono con i centri di rotazione anatomici né meccanici, rendendo impossibile una mappatura diretta sui 29 DoF dell'umanoide;

2. **Onere di ri-addestramento o adattamento euristico**: Per tentare di colmare questo disallineamento all'interno della toolchain, si sarebbe dovuto intraprendere un complesso lavoro di fine-tuning delle reti neurali 2D (per costringerle a identificare punti conformi alle cerniere meccaniche del robot) oppure sviluppare complessi strati intermedi di adattamento geometrico manuale, vanificando la flessibilità e la riproducibilità industriale del sistema.

2.3.2 Varianza delle lunghezze ossee, assenza di vincoli $SO(3)$ e shoulder popping

Poiché la triangolazione DLT opera in modo puramente geometrico e indipendente su ciascun frame temporale, la distanza euclidea tra landmark consecutivi (ad esempio la distanza tra il gomito e il polso o tra anca e ginocchio) non risulta vincolata ad alcun modello scheletrico rigido:

- **Varianza delle lunghezze ossee (bone length variance)**: In presenza di piccole occlusioni o imprecisioni di stima 2D, la distanza tra i punti varia sensibilmente da un istante al successivo, generando un fenomeno di "ossa elastiche";
- **Collasso e aberrazioni dimensionali dello scheletro**: Nei frame caratterizzati da perdita temporanea di tracciamento o allucinazione di un landmark su elementi di sfondo, la struttura scheletrica subisce distorsioni geometriche catastrofiche. Come documentato nel modello BVH visualizzato in Blender (Figura 2.16), i segmenti corporei assumono lunghezze non fisiche, con collasso delle proporzioni anatomiche e deformazione dei volumi corporei;
- **Instabilità nella cinematica inversa (shoulder popping)**: Quando si tenta di calcolare le rotazioni angolari di giunto tramite cinematica inversa numerica partendo da segmenti a lunghezza variabile, i solutori incappano in singolarità matematiche. Il risultato si manifesta in rotazioni incontrollate, torsioni innaturali e violenti scatti articolari della spalla (fenomeno noto come shoulder popping), del tutto incompatibili con i limiti di coppia, velocità ed escursione articolare di un vero robot.

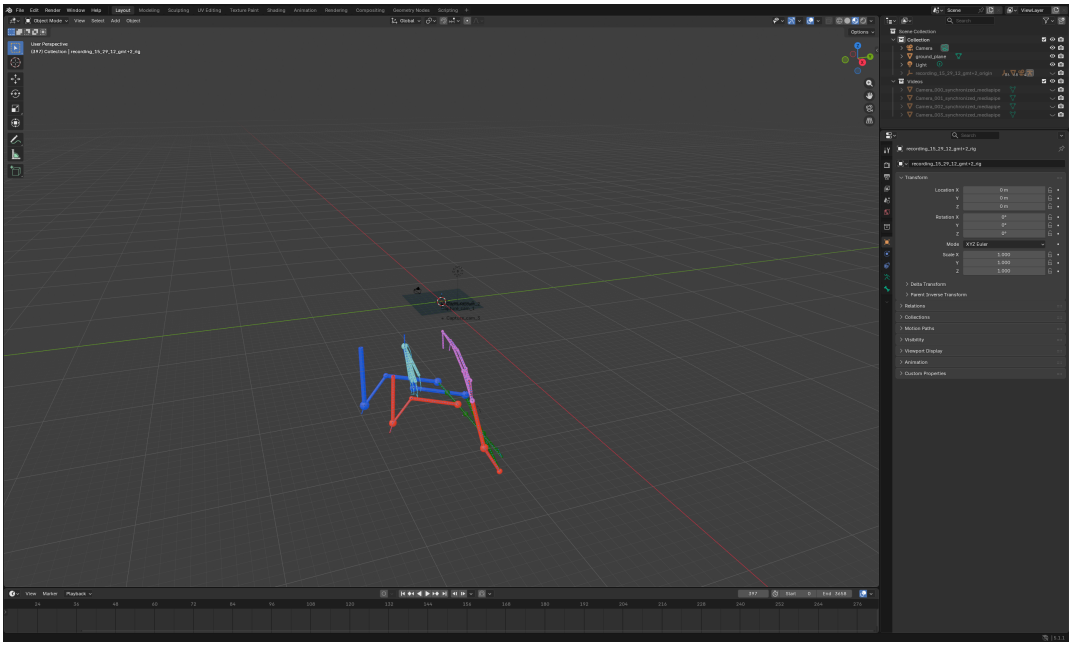


Figura 2.16: Visualizzazione dell'armature BVH su Blender a seguito del collasso del tracciamento in FreeMoCap: deformazione catastrofica delle proporzioni scheletriche, alterazione delle dimensioni dei segmenti ossei e divergenza geometrica.

2.3.3 Motivazione del cambio di paradigma tecnologico

L'evidenza sperimentale raccolta ha dimostrato in modo inequivocabile che, per abilitare un trasferimento affidabile del movimento verso un robot umanoide, l'approccio basato su

mocap ottico a triangolazione geometrica sparsa presenta limiti strutturali non superabili con semplici filtri a valle.

Si è resa quindi necessaria una radicale revisione metodologica: abbandonare la triangolazione di punti cartesiani discreti per orientarsi verso un modello neurale parametrico vincolato (come NVIDIA GEM-X con modello corporeo SOMA), capace di stimare direttamente rotazioni $SO(3)$ regolarizzate e lunghezze ossee invarianti a partire da singoli flussi video monoculari. Questa transizione tecnologica ha imposto a sua volta una completa reingegnerizzazione dell'infrastruttura software di supporto, oggetto del capitolo successivo.

Capitolo 3

Architettura Software, Gestione delle Dipendenze ed Ecosistema Modulare

Il presente capitolo espone l'infrastruttura ingegneristica e l'architettura software modulare concepita per orchestrare l'intera pipeline di trasferimento del moto umano sul robot umanoide Unitree G1. Viene analizzato in dettaglio il fenomeno del "Dependency Hell" che caratterizza gli odierni stack di Physical AI, originato dalla coesistenza di librerie grafiche legacy, framework di deep learning per computer vision, kernel differenziabili su GPU e motori fisici basati su Pixar OpenUSD. Per garantire stabilità operativa, isolamento dei guasti e perfetta riproducibilità nel reparto R&D aziendale, è stata implementata una strategia a quattro ambienti virtuali Conda ermetici (env_freemocap, env_gemx, env_soma, env_isaaclab). Vengono quindi formalizzati i protocolli di interscambio dati basati rigorosamente su standard aperti (.mp4, .bvh/.npy, .csv, .usd/.onnx), evidenziando i benefici architetturali in termini di manutenibilità, sostituibilità plug-and-play dei singoli stadi e portabilità del know-how.

3.1 Risoluzione dei Conflitti Software e Strategia degli Ambienti Isolati

La transizione da una ripresa video 2D non calibrata fino all'esecuzione di una policy di controllo dinamico in simulazione richiede la convergenza di una molteplicità di framework tecnologici eterogenei, sviluppati da consorzi industriali e gruppi di ricerca differenti. L'integrazione di tali componenti all'interno di una workstation di calcolo aziendale impone una rigorosa analisi delle dipendenze di sistema e l'adozione di un'architettura software ad ambienti ermeticamente isolati.

3.1.1 Il problema del Dependency Hell nella robotica moderna

Nel contesto dello sviluppo software per robotica umanoide e intelligenza artificiale fisica (Physical AI), l'integrazione monolitica di tutti i pacchetti all'interno di un unico ambiente globale è divenuta un'impossibilità tecnica. Questo fenomeno, comunemente noto in ingegneria del software come Dependency Hell, si manifesta a causa di quattro fattori strutturali:

1. **Conflitti di runtime CUDA e versioni PyTorch**: I modelli di visione neurale per la stima della posa, quali GEM-X [14] e ViTPose [33], si basano sul framework PyTorch [24] e richiedono estensioni binarie C++/CUDA pre-compilate (ad esempio il toolkit MMCV). Tali librerie sono compilate rigidamente contro versioni specifiche del CUDA Toolkit (ad esempio CUDA 12.1) e dell'interprete Python, risultando binaricamente incompatibili con release differenti;

2. **Kernel differenziabili compilati al volo (JIT) su GPU**: Il modulo di retargeting cinematico SOMA Retargeter [23] impiega il compilatore NVIDIA Warp [21], il quale genera kernel C++/CUDA tramite LLVM in fase di esecuzione (Just-In-Time). Questo paradigma necessita di header e driver API specifici che possono collidere con i wrapper di runtime di altri framework;

3. **Interpreti embedded e runtime OpenUSD / PhysX 5**: I moderni simulatori di robotica avanzata, in particolare NVIDIA Isaac Sim [6] e Isaac Lab [18], incorporano un interprete Python embedded strettamente collegato alle librerie binarie C++ di Pixar OpenUSD [25] e del motore fisico PhysX 5. L'installazione di librerie esterne non conformi all'Application Binary Interface (ABI) di Omniverse provoca corruzioni di memoria e crash di sistema immediati;

4. **Stack grafici legacy e multiprocessing su CPU**: Piattaforme di calibrazione multi-camera tradizionali come FreeMoCap [17] fanno affidamento su librerie GUI desktop e versioni di MediaPipe [16] ottimizzate per CPU e pipeline sincrone, le cui dipendenze di packaging entrano in conflitto diretto con i driver GPU di ultima generazione.

La coesistenza forzata di questi moduli in un unico spazio di librerie condivise condurrebbe a sovrascritture di simboli dinamici e a un'incontrollabile instabilità del sistema.

3.1.2 Ambienti virtuali standard (`venv`) vs ecosistemi Conda: gestione dei pacchetti e file di configurazione (`.txt` vs `.yaml`)

Nei repository open-source distribuiti su piattaforme di controllo versione (quali GitHub), gli sviluppatori rilasciano file di configurazione formali atti a congelare l'albero delle dipendenze per consentire la riproduzione deterministica dell'ambiente di calcolo. Esiste tuttavia una profonda disparità metodologica ed operativa tra l'impiego degli ambienti virtuali standard di Python (`venv`) e gli ecosistemi binari Conda [1]:

- **Ambienti virtuali Python (`venv` con gestore `pip`)**: `venv` isola esclusivamente i moduli Python distribuiti dal repository centrale PyPI (pacchetti wheel `.whl` o archivi sorgente). Il suo limite intrinseco risiede nell'impossibilità di incapsulare l'interprete Python stesso (il quale viene ereditato dall'installazione globale del sistema operativo host) e nell'incapacità di gestire dipendenze binarie native di basso livello, quali librerie C/C++, header di sistema e runtime CUDA Toolkit. Il file di specifica associato è il tradizionale `requirements.txt`, il quale elenca linearmente i package e le rispettive versioni con vincoli di uguaglianza stretta (`pacchetto==1.2.3`). L'esportazione automatica dello stato attivo avviene mediante il comando:

  ```
  pip freeze > requirements.txt
  ```

  mentre il deployment su una nuova macchina richiede l'esecuzione di:

  ```
  pip install -r requirements.txt
  ```

- **Ambienti isolati Conda (`conda`)**: Conda è un gestore di pacchetti e ambienti binari multi-linguaggio (cross-language binary package manager). Esso consente di incapsulare nello stesso ambiente isolato non soltanto i pacchetti Python, ma la versione esatta dell'interprete Python desiderato (ad esempio Python 3.9 in un modulo e Python 3.10 in un altro), i compilatori di supporto, le librerie C/C++ e l'intera suite binaria del CUDA Toolkit (`cudatoolkit`, `cudnn`). L'ecosistema di distribuzione Anaconda include nativamente una ricca suite di librerie ottimizzate per il calcolo numerico, l'algebra lineare e la ricerca scientifica (quali `numpy`, `scipy`, `matplotlib` e `pandas`), fornendo una solida base per la manipolazione preliminare dei dati.

  L'interfaccia principale per l'interazione con l'ecosistema è il terminale specializzato Anaconda Prompt, accessibile dal menu di sistema (Figura 3.1a). All'avvio, la riga

di comando antepone visivamente tra parentesi tonde il nome dell'ambiente attivo, ad esempio `(base)` o `(env_nome)` (Figura 3.1b), garantendo all'operatore la certezza del contesto di esecuzione.

I tre comandi essenziali per la gestione operativa sono:

```
# 1. Creazione di un ambiente pulito con versione Python
   dedicata
conda create -n nome_ambiente python=3.10

# 2. Ispezione della lista degli ambienti configurati sulla
   macchina
conda env list

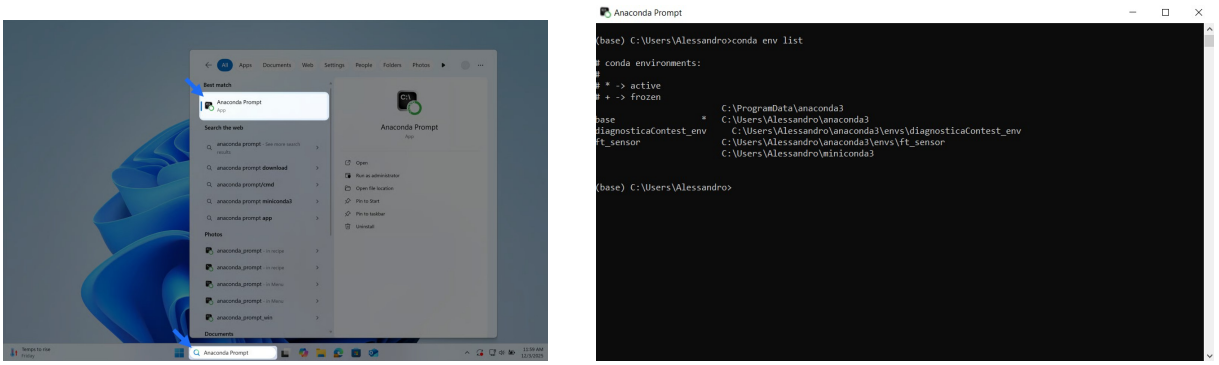
# 3. Attivazione dell'ambiente di lavoro
conda activate nome_ambiente
```

Il file di configurazione dichiarativo di riferimento è `environment.yaml` (o `.yml`), strutturato gerarchicamente per specificare il nome dell'ambiente, i canali ufficiali di download (channels) e le dipendenze native. L'esportazione automatica dello stato attivo avviene con:

```
conda env export > environment.yaml
```

mentre la ricreazione deterministica da repository si effettua con:

```
conda env create -f environment.yaml
```



(a) Accesso ad Anaconda Prompt dal menu di sistema.

(b) Interfaccia CLI del terminale Conda con evidenza del prompt dell'ambiente attivo tra parentesi.

Figura 3.1: Interfaccia di sistema e sessione interattiva da riga di comando per la gestione degli ambienti Conda sulla workstation.

In Tabella 3.1 viene formalizzato il confronto sinottico tra le due metodologie di isolamento.

Caratteristica	**Python Virtual Environment (`venv`)**	**Conda Environment (`conda`)**
File di configurazione	`requirements.txt`	`environment.yaml` / `.yml`
Isolamento interprete Python	Non supportato (ereditato da OS)	Supportato nativamente per ambiente
Gestione librerie C/C++ e ABI	Non supportata (richiede build esterne)	Pacchetti binari pre-compilati isolati
Gestione CUDA Toolkit nativo	Limitata (solo binding wheel PyTorch)	Completa (`cudatoolkit`, `cudnn`, header)
Comando esportazione automatica	`pip freeze > requirements.txt`	`conda env export > environment.yaml`
Comando installazione/deploy	`pip install -r requirements.txt`	`conda env create -f environment.yaml`

Tabella 3.1: Confronto metodologico tra ambienti virtuali Python (`venv`) ed ecosistemi Conda (`conda`).

3.1.3 Mappatura e isolamento dei 4 ambienti Conda dedicati

Alla luce delle incompatibilità evidenziate, l'intera pipeline di ricerca e sviluppo è stata strutturata suddividendo le quattro fasi operative in altrettanti ambienti Conda reciprocamente isolati, descritti in Tabella 3.2.

Ambiente Conda	Python	Librerie Principali	Ruolo nella Pipeline
`env_freemocap`	3.9 / 3.10	FreeMoCap, OpenCV, MediaPipe	Acquisizione multi-camera e benchmark ottico DLT
`env_gemx`	3.10	PyTorch (CUDA 12.x), GEM-X, ViTPose	Stima 3D della posa parametrica SOMA da video 2D
`env_soma`	3.10	SOMA Retargeter, NVIDIA Warp	Kinematic Retargeting accelerato su GPU per Unitree G1
`env_isaaclab`	3.10 (Omni)	NVIDIA Isaac Lab, PhysX 5, OpenUSD	Simulazione dinamica e PoC di Reinforcement Learning

Tabella 3.2: Mappatura dei quattro ambienti Conda isolati costituenti l'infrastruttura software modulare.

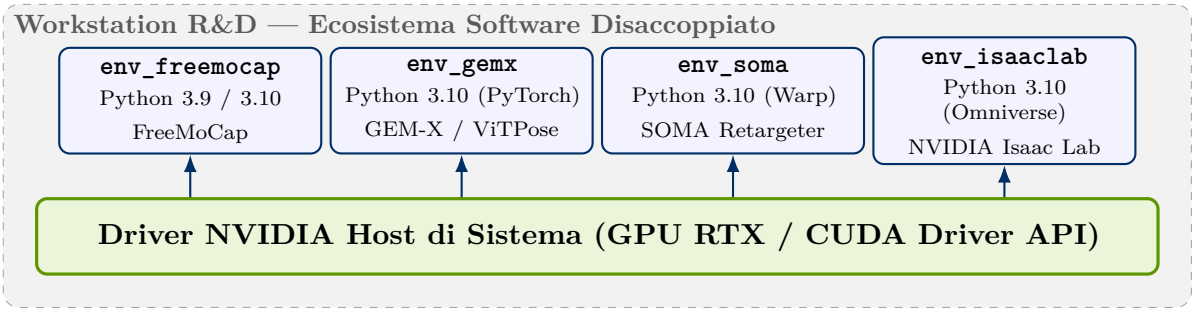


Figura 3.2: Diagramma architetturale dell'isolamento dei quattro ambienti Conda basati su un unico driver GPU host.

3.1.4 Gestione dei conflitti tra driver CUDA, versioni PyTorch e OpenUSD

L'adozione della topologia a quattro ambienti ha risolto in modo definitivo i potenziali colli di bottiglia e i conflitti di compatibilità di basso livello:

- **Disaccoppiamento tra Driver Host e CUDA Toolkit**: A livello di sistema operativo host è installato un driver NVIDIA aggiornato in grado di supportare l'interfaccia driver CUDA 12.x. Ciascun ambiente virtuale incapsula in modo indipendente la versione di CUDA Toolkit (e i relativi binari cuDNN / TensorRT) richiesta dal proprio framework, evitando che l'aggiornamento di un pacchetto di visione comprometta il simulatore;
- **Segregazione dell'ABI C++ di Omniverse**: Il runtime di Isaac Sim carica una suite di librerie dinamiche OpenUSD compilate con specifici flag di ottimizzazione

C++. La segregazione in *env_isaaclab* impedisce che moduli di deep learning esterni vadano a sovrascrivere i puntatori ai simboli dinamici di PhysX 5;

- **Isolamento della cache dei kernel Warp JIT**: L'ambiente *env_soma* isola la cartella di compilazione e caching dei kernel LLVM di NVIDIA Warp, azzerando le interferenze con i grafi di calcolo eseguiti da PyTorch in *env_gemx*.

3.2 Protocolli di Interscambio Dati e Pipeline Orchestration

Il corretto funzionamento di un'architettura software a moduli disaccoppiati richiede che il passaggio di informazioni tra i vari stadi avvenga attraverso protocolli di interscambio standardizzati e formati di file universali e aperti.

3.2.1 Flusso di interscambio dati end-to-end e formati di comunicazione

Il flusso operativo dell'infrastruttura proposta si articola attraverso una catena sequenziale di tre moduli di elaborazione disaccoppiati, intervallati da formati di file standard aperti che fungono da contratti di interfaccia immutabili (Figura 3.3):

1. **Input Iniziale — Flusso Video 2D (`.mp4`)**: Il punto di partenza dell'intera catena è una registrazione video RGB bidimensionale non calibrata dell'operatore umano in ambiente di lavoro (acquisita tramite webcam commerciale o smartphone);

2. **Stadio 1 — Stima Neurale della Posa 3D (*env_gemx*)**: Il modello di visione GEM-X elabora il video 2D ed estrae la cinematica tridimensionale del corpo umano secondo il modello corporeo SOMA. L'output viene serializzato nei formati aperti `.bvh` (gerarchia scheletrica) e `.npy` (tensori delle rotazioni spaziali e traiettoria globale del baricentro);

3. **Stadio 2 — Kinematic Retargeting su GPU (*env_soma*)**: Il solutore SOMA Retargeter ingerisce i file di posa `.bvh`/`.npy` e risolve la cinematica inversa vincolata (QP-IK) per i gradi di libertà del robot Unitree G1. L'output consiste in serie temporali articolari tabulari in formato standard `.csv` (corredate da un video MP4 di validazione visuale con sovrapposizione scheletrica);

4. **Stadio 3 — Simulazione Dinamica e Policy Learning (*env_isaaclab*)**: All'interno dell'ambiente di simulazione su GPU (Isaac Lab), il modello dinamico del robot in formato OpenUSD (`.usd`) viene istanziato con parametri PhysX 5 e guidato dalle

traiettorie `.csv` per l'addestramento e la validazione del tracking in Reinforcement Learning, producendo come output finale la policy neurale serializzata (`.pt` o `.onnx`).

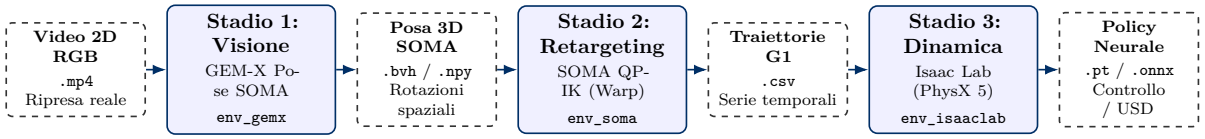


Figura 3.3: Diagramma di flusso logico e protocolli di interscambio dati tra i tre stadi computazionali della pipeline.

3.2.2 Manutenibilità, modularità e aggiornabilità dei singoli stadi

La separazione a stadi garantita da formati aperti apporta vantaggi decisivi in un'ottica di ingegneria industriale:

- **Isolamento dei guasti**: Un'anomalia o un'eccezione numerica manifestatasi durante la stima della posa (ad esempio un frame video eccessivamente sfocato o occluso) interrompe unicamente lo stadio a monte in *env_gemx*, senza provocare instabilità o corruzioni di memoria nel simulatore o nel retargeter;
- **Principio Plug-and-Play e sostituibilità tecnologica**: Ciascun componente può essere aggiornato o sostituito in modo trasparente:
  - Il modulo di estrazione del movimento da video (GEM-X) potrà essere affiancato da modelli generativi basati su diffusione cinematica (ad esempio NVIDIA Kimodo [26]) mantenendo inalterato il solutore di retargeting SOMA, purché l'output rispetti la convenzione del formato parametrico di scambio;
  - Il simulatore dinamico (Isaac Lab) potrà essere sostituito o affiancato da framework Sim-to-Sim alternativi (quali ProtoMotions [30] o MuJoCo [31]), i quali possono ingerire direttamente le medesime serie temporali articolari registrate nel file `.csv`;
- **Parallelismo di sviluppo aziendale**: Differenti figure professionali all'interno del reparto R&D possono sviluppare, testare e ottimizzare in parallelo il solutore di retargeting e la funzione di reward del Reinforcement Learning senza generare conflitti sulle versioni delle librerie.

3.2.3 Riproducibilità scientifica ed esportazione delle configurazioni `environment.yaml`

Per assicurare che l'infrastruttura tecnologica sviluppata costituisca un patrimonio aziendale persistente e trasferibile, tutti gli ambienti sono stati formalizzati mediante file descrittivi `environment.yaml`.

Nella Lista 3.1 viene riportata la struttura dichiarativa standard impiegata per l'ambiente di stima della posa neurale *env_gemx*. L'esplicitazione dei canali ufficiali e il congelamento delle release garantiscono che la creazione dell'ambiente su nuove workstation di calcolo o nodi cluster avvenga in modo trasparente e con identico comportamento deterministico.

```
name: env_gemx
channels:
  - pytorch
  - nvidia
  - conda-forge
  - defaults
dependencies:
  - python=3.10
  - pytorch=2.1.0
  - torchvision=0.16.0
  - pytorch-cuda=12.1
  - numpy>=1.24.0
  - scipy>=1.10.0
  - pip
  - pip:
      - mmcv==2.1.0
      - mmpose==1.3.1
      - onnxruntime-gpu==1.17.0
      - opencv-python-headless>=4.8.0
```

Lista 3.1: Esempio di struttura dichiarativa del file `environment.yaml` per l'ambiente *env_gemx*.

3.3 Quadro di Sintesi dell'Ecosistema Software

A conclusione della trattazione architetturale, in Tabella 3.3 viene riportato il quadro sinottico dell'interoperabilità software complessiva, riassumendo gli ambienti Conda, i dati di ingresso e uscita, le risorse computazionali impiegate e gli standard aperti adottati per la pipeline.

Stadio	**Ambiente**	**Input**	**Output**	**Hardware**	**Standard**
0. Benchmark	`env_freemocap`	Video Multi-Cam	BVH grezzo, CSV landmark	CPU Multi-Core	BVH non vincolato
1. Visione 3D	`env_gemx`	Video 2D (`.mp4`)	Posa SOMA 3D (`.bvh`, `.npy`)	GPU Tensor Cores	Rig parametrico SOMA
2. Retargeting	`env_soma`	Posa SOMA 3D	Traiettorie G1 (`.csv`, `.mp4`)	GPU Warp JIT	Serie temporali (`.csv`)
3. Simulazione	`env_isaaclab`	CSV + USD G1	Policy (`.pt`, `.onnx`), USD	GPU VRAM (PhysX 5)	Pixar OpenUSD / PyTorch

Tabella 3.3: Matrice sinottica di interoperabilità dell'ecosistema software modulare.

La solida infrastruttura ingegneristica definita nel presente capitolo costituisce il prerequisito abilitante per la trattazione analitica e matematica dei moduli di stima monoculare GEM-X e di retargeting cinematico SOMA, presentati nel Capitolo 4.

Capitolo 4

Visione Monoculare Neurale e Kinematic Retargeting con GEM-X e SOMA

Il presente capitolo espone l'architettura tecnica, matematica e algoritmica della pipeline proposta per superare strutturalmente le limitazioni del motion capture multi-camera tradizionale riscontrate nel Capitolo 2. La catena di elaborazione riceve in ingresso un singolo flusso video RGB bidimensionale non calibrato dell'operatore ed estrae la cinematica tridimensionale completa del corpo umano in coordinate assolute (World-Space) tramite il modello neurale generalista GEM-X, basato sul framework parametrico SOMA a 77 giunti. Successivamente, il modulo SOMA Retargeter, accelerato su GPU mediante kernel differenziabili in NVIDIA Warp, risolve un problema di ottimizzazione cinematica inversa vincolata per mappare il moto umano sui 29 gradi di libertà del robot umanoide Unitree G1, garantendo il rispetto rigoroso dei limiti articolari, la preservazione dei contatti al suolo e l'esportazione di serie temporali tabulari in formato .csv pronte per la simulazione dinamica e il controllo.

4.1 Estrazione del Movimento con GEM-X

Il superamento dell'infrastruttura di cattura multi-camera basata su FreeMoCap ha richiesto un cambio di paradigma verso la stima della posa monoculare basata su modelli neurali di tipo foundation [14]. L'impiego di una singola telecamera RGB commerciale (o di uno smartphone) consente di estrarre la cinematica dell'operatore in contesti industriali reali, eliminando i vincoli di calibrazione estrinseca con scacchiere ottiche e la complessità legata alla sincronizzazione simultanea dei flussi video di telecamere multiple (processo che in sistemi multi-vista come FreeMoCap richiede la gestione coordinata dei trigger di avvio

e dei buffer di acquisizione per evitare disallineamenti temporali). Come introdotto nella visione d'insieme della pipeline software (Figura 1.2), questo modulo rappresenta il primo stadio critico per trasformare il dato visivo 2D non calibrato in cinematica tridimensionale biologicamente plausibile.

4.1.1 Acquisizione Video 2D e Scenario Operativo

La fase di ingresso della pipeline accetta filmati video standard registrati con singola inquadratura frontale o a tre quarti del soggetto ($I_t \in \mathbb{R}^{H \times W \times 3}$). L'invarianza del modello rispetto alla focale e ai parametri intrinseci consente di elaborare sequenze video non strutturate in ambienti reali, inclusi video tutorial registrati in officina o riprese di linee di montaggio.

La sequenza di fotogrammi viene processata all'interno dell'ambiente isolato `env_gemx` attraverso una cascata algoritmica a tre stadi principali, formalizzata nello schema essenziale di Figura 4.1:

1. **Fase 1: Rilevamento e Tracking 2D**: individuazione del bounding box con YOLOX e associazione temporale con ByteTrack;
2. **Fase 2: Stima della Posa 2D**: estrazione delle heatmap dei 77 keypoint canonici SOMA tramite ViTPose;
3. **Fase 3: 3D Lifting e Temporal Fusion**: sollevamento tridimensionale, predizione delle rotazioni articolari in $SO(3)$ e disaccoppiamento del moto della telecamera nello spazio assoluto (World-Space).

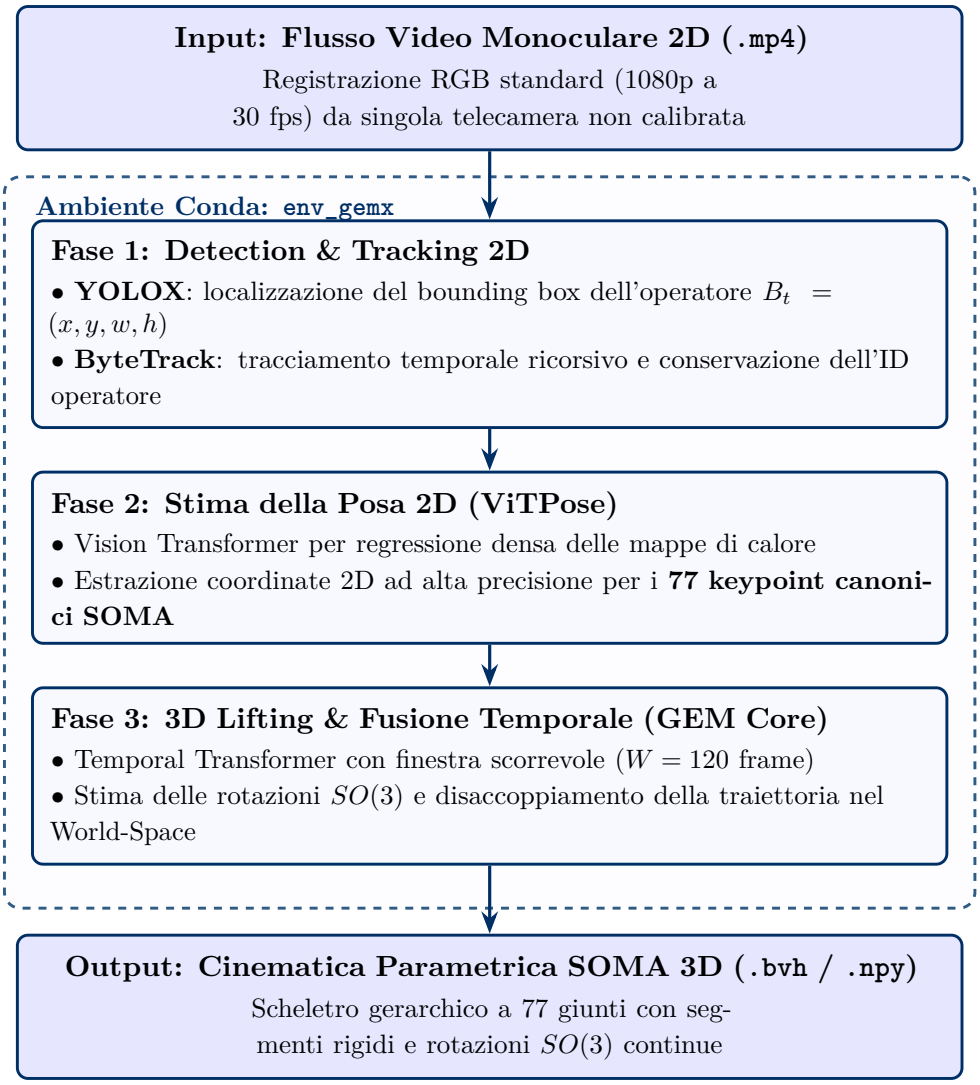


Figura 4.1: Architettura a cascata della pipeline di visione neurale monoculare GEM-X all'interno dell'ambiente isolato `env_gemx`.

4.1.2 Architettura a Stadi: YOLOX, ByteTrack e ViTPose a 77 Keypoint

L'estrazione della cinematica corporea bidimensionale adotta il paradigma del Tracking-by-Detection, separando la localizzazione visiva dall'inseguimento temporale in due ruoli complementari:

- **Rilevamento visivo istantaneo (YOLOX)**: analizza i pixel del singolo fotogramma per individuare la figura umana. Risponde alla domanda: «Dove si trova l'operatore e qual è il suo riquadro di delimitazione?» (operando senza memoria storica);
- **Associazione temporale (ByteTrack)**: riceve i riquadri e li collega nel tempo. Risponde alla domanda: «Il riquadro corrente appartiene allo stesso operatore

tracciato nei frame precedenti?».

A questi moduli si affianca la rete neurale profonda ViTPose per l'estrazione fine dei punti scheletrici. Nel dettaglio:

1. Rilevamento del Soggetto con YOLOX. L'algoritmo di localizzazione visiva YOLOX [7] individua la porzione dell'immagine contenente la figura dell'operatore. A differenza dei rilevatori tradizionali vincolati a forme e riquadri prefissati (ancore), l'approccio privo di ancore predice direttamente il centro e l'estensione del corpo a partire dalle caratteristiche visive. Questa flessibilità permette al modello di riconoscere posture corporee insolite o fortemente inclinate (come l'operatore chino sul banco o parzialmente coperto da macchinari). Inoltre, la struttura disaccoppiata della testa neurale separa il riconoscimento della persona dalla stima precisa dei suoi confini, garantendo una localizzazione affidabile anche in ambienti industriali complessi.

2. Tracciamento Temporale e Persistenza dell'Identità con ByteTrack. L'algoritmo ByteTrack [34] associa i riquadri istante per istante, prevenendo perdite di traccia e scambi di identità dovuti a movimenti rapidi o ad attrezzi che nascondono temporaneamente parti del corpo. La continuità del tracciamento è garantita da una strategia su due livelli:

- **Filtro di Kalman**: stima la posizione attesa dell'operatore in base alla velocità e alla direzione nei fotogrammi precedenti;
- **Associazione a doppio stadio**: abbina prima i rilevamenti più nitidi e certi, e recupera successivamente quelli a confidenza inferiore (es. quando l'operatore è parzialmente coperto da un macchinario), evitando buchi temporali e scartando i falsi positivi dello sfondo.

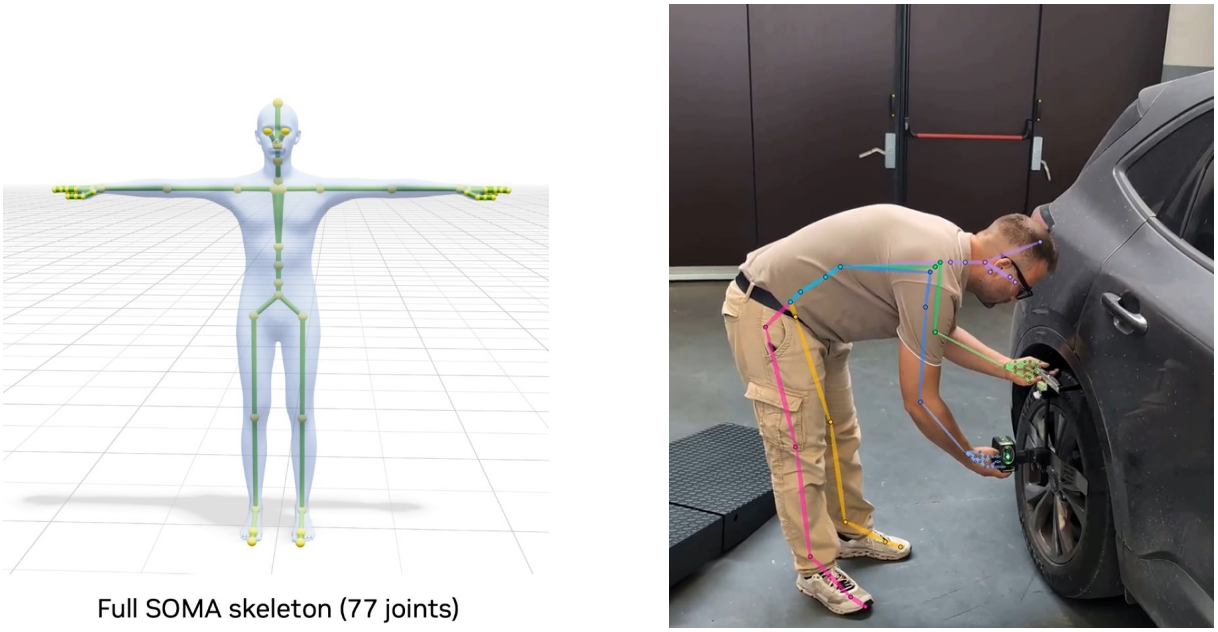
Questo assicura che il ritaglio corporeo fornito a ViTPose appartenga ininterrottamente allo stesso operatore, isolandolo da eventuali altre persone presenti nell'ambiente.

3. Regressione dei Keypoint con ViTPose Fine-Tuned a 77 Giunti. Il ritaglio normalizzato della figura umana viene alimentato alla rete Vision Transformer ViTPose [33]. Nella sua configurazione standard di letteratura (addestrata su benchmark come MS COCO), ViTPose stima soltanto i 17 punti chiave principali del corpo. All'interno della pipeline GEM-X, la testa di uscita del modello è stata riconfigurata a 77 canali paralleli e riaddestrata su dataset a corpo intero (Whole-Body Mocap).

A differenza di soluzioni modulari come MediaPipe, che impiegano tre reti disgiunte per corpo, mani e volto (generando frequenti disallineamenti cinematici a livello dei polsi e del collo), ViTPose estrae l'intera topologia unificata dello standard SOMA [28] in un unico passaggio neurale:

- **Corpo Principale (23 giunti)**: pelvi, colonna vertebrale (lombare, toracica, cervicale), spalle, gomiti, polsi, anche, ginocchia e caviglie;
- **Mani a 5 Dita (30 giunti totali)**: 15 punti articolari per ciascuna mano (metacarpi e falangi), fondamentali per le prese;
- **Piedi e Caviglie (6 giunti)**: tallone, tarso e alluce per lato, critici per l'appoggio a terra;
- **Testa ed Espressività Facciale (18 giunti)**: orientamento del cranio, sguardo e punti mandibolari.

In Figura 4.2 viene illustrato il confronto tra la struttura teorica dello scheletro SOMA e la sua corrispondente estrazione sull'operatore durante una lavorazione in officina.



(a) Modello teorico a 77 keypoint.

(b) Rilevamento reale sull'operatore.

Figura 4.2: Scheletro SOMA a 77 keypoint: (a) modello teorico e (b) stima sull'operatore durante una prova in officina (fonte per (a): [28]).

4.1.3 GEM Core Temporal Transformer e Separazione del Movimento

Il passaggio dalla posa bidimensionale dei singoli fotogrammi allo scheletro tridimensionale (3D Lifting) è governato dal modulo GEM Core Temporal Transformer [14]. L'architettura analizza una finestra temporale scorrevole di circa 4.0 secondi (120 fotogrammi a 30 fps), sfruttando meccanismi di auto-attenzione per comprendere la continuità e la dinamica del gesto.

Per evitare che le oscillazioni della telecamera o i movimenti di chi effettua la ripresa distorcano la cinematica, la rete neurale separa in modo rigoroso il moto dell'operatore in due componenti complementari:

- **1. Posa articolare locale**: definisce come si muovono e ruotano le singole articolazioni dello scheletro SOMA (braccia, gambe, colonna vertebrale e dita) le une rispetto alle altre, catturando la forma del gesto umano indipendentemente dalla prospettiva di ripresa;
- **2. Spostamento globale del corpo**: traccia la posizione e l'orientamento assoluto del baricentro dell'operatore rispetto al pavimento dell'ambiente di lavoro.

Questo disaccoppiamento consente di isolare i tremolii o i micromovimenti della telecamera dalla cinematica reale della persona: la traiettoria del baricentro risulta così fluida, coerente con il piano di calpestio e priva di scivolamenti artificiali sul suolo.

4.1.4 Finestra Temporale, Dinamica del Gesto e Strategia Operativa

L'accuratezza della ricostruzione tridimensionale è strettamente legata alla frequenza di campionamento della telecamera e alla dinamica con cui l'operatore esegue le mansioni.

Risoluzione Temporale della Finestra Scorrevole. Il modello GEM Core elabora il flusso video attraverso una finestra scorrevole di $W = 120$ fotogrammi, che a una frequenza standard di 30 fps corrisponde a un intervallo temporale continuo di circa 4.0 secondi. Per consentire al meccanismo di auto-attenzione di cogliere l'evoluzione temporale del gesto, è opportuno che ogni singola azione (ad esempio l'avvicinamento dell'arto o la presa di un utensile) sia descritta da un numero sufficiente di frame e non si esaurisca in scatti eccessivamente repentini.

Effetto del Motion Blur. Movimenti manuali troppo rapidi generano sfocature da movimento (motion blur) nei singoli fotogrammi registrati con telecamere commerciali a 30 fps. Tale sfocatura riduce la nitidezza dei dettagli visivi, rendendo più difficoltosa e rumorosa l'individuazione della posizione precisa delle dita e dei giunti articolari da parte della rete di visione.

Strategia Operativa di Dimostrazione Controllata. Per superare questa criticità senza dover ricorrere a costosi sensori industriali ad altissimo frame rate o con otturatore globale (global shutter), si è adottata una linea guida operativa empirica durante le fasi di ripresa:

- L'operatore viene istruito a eseguire le lavorazioni con un ritmo cadenzato, naturale e fluido, evitando accelerazioni brusche o scatti improvvisi;
- I gesti di manipolazione vengono articolati con continuità, permettendo alla finestra temporale di osservare l'intera transizione dello stato cinematico.

Questo accorgimento metodologico consente di ottenere sequenze tridimensionali stabili e prive di rumore impulsivo impiegando semplici telecamere RGB commerciali a 30 fps, rendendo la pipeline accessibile, economica e immediatamente replicabile in contesti di officina reale.

4.1.5 Risoluzione dei Difetti del BVH: Rigidità dello Scheletro e Continuità del Moto

L'adozione del modello parametrico SOMA [28], che estende l'eredità e supera le limitazioni dei modelli corporei parametrici classici quali SMPL [15], all'interno di GEM-X risolve alla radice i due limiti strutturali che avevano compromesso i file di movimento generati da FreeMoCap (descritti nella Sezione 2.3):

1. Costanza delle Lunghezze Corporee. Mentre nei metodi basati su landmark indipendenti la triangolazione ottica provocava continue variazioni nella lunghezza degli arti (il fenomeno del bone stretching), SOMA modella il corpo umano come una catena cinematica rigida ad albero. La distanza tra i giunti (omero, avambraccio, coscia, dita) viene fissata a monte e rimane rigorosamente costante per tutta la durata del filmato, annullando ogni artefatto a fisarmonica.

2. Rotazioni Spaziali Continue. Anziché produrre semplici coordinate cartesiane isolate, GEM-X stima direttamente l'orientamento e le rotazioni spaziali relative per ciascuno dei 77 giunti dello scheletro. Questo approccio garantisce traiettorie angolari morbide e continue nel tempo, prevenendo blocchi cinematici ed eliminando le repentine inversioni di orientamento o scatti articolari delle braccia.

4.2 Integrazione GEM-X / SOMA e Retargeting su Unitree G1

Uno dei punti di forza dell'architettura adottata risiede nella stretta interoperabilità tra i moduli: GEM-X e SOMA Retargeter sono stati progettati per lavorare in perfetta sinergia attorno alla topologia condivisa dello scheletro SOMA. All'interno della pipeline è pertanto possibile attivare l'esecuzione diretta del retargeting a valle dell'estrazione tridimensionale, convertendo il flusso video direttamente nelle traiettorie angolari per il robot.

Nel passaggio dal corpo umano alla struttura del robot umanoide Unitree G1 (29 gradi di libertà: 12 per le gambe, 14 per le braccia e 3 per il busto), il sistema esegue preventivamente una riscalatura proporzionale dei segmenti scheletrici basata sulle dimensioni nominali del file URDF (Figura 4.3). Questo passaggio compensa la differenza di statura tra l'operatore (1.70–1.85 m) e il robot (1.32 m), evitando estensioni innaturali degli arti o singolarità cinematiche.

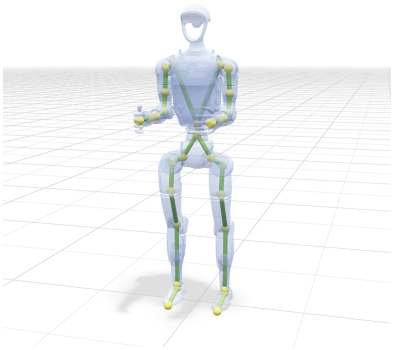
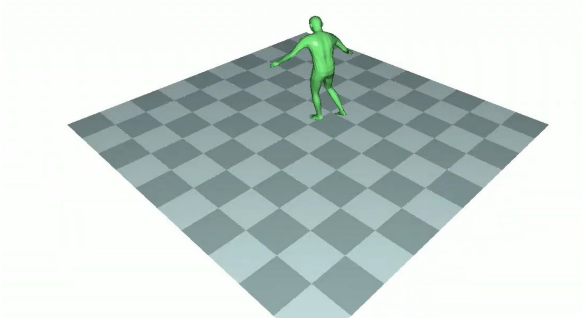


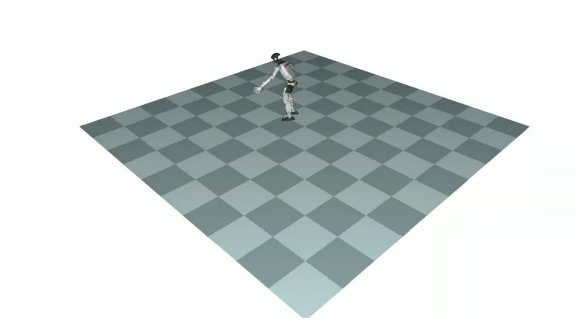
Figura 4.3: Rappresentazione scheletrica per il retargeting sui 29 gradi di libertà dell'Unitree G1 (fonte: [23]).

Il solutore calcola in tempo reale su GPU la configurazione ottimale dei 29 motori del robot, assicurando che i terminali (mani e piedi) e gli orientamenti degli arti seguano fedelmente il gesto umano. Al tempo stesso, blocca rigidamente la pianta del piede al suolo durante la fase di appoggio per eliminare ogni forma di scivolamento (foot skating) e garantisce il pieno rispetto dei finecorsa meccanici e delle velocità limite dei motori.

Qualora si utilizzi la modalità batch automatica, SOMA Retargeter genera in uscita due filmati renderizzati globali di controllo (Figura 4.4), che consentono di ispezionare separatamente la cinematica tridimensionale dello scheletro umano SOMA e la corrispondente traiettoria spaziale ricalcolata per l'Unitree G1 prima del passaggio in simulazione dinamica.



(a) Traiettoria globale dello scheletro umano SOMA.



(b) Traiettoria globale ricalcolata per Unitree G1.

Figura 4.4: Render globali di controllo generati da SOMA Retargeter: (a) scheletro umano SOMA e (b) cinematica ricalcolata per Unitree G1.

4.3 Validazione Sperimentale e Ispezione delle Traiettorie

Per accertare che la pipeline fosse realmente solida e affidabile al di fuori dei contesti teorici ideali, è stata condotta una campagna di prove pratiche e verifiche visive direttamente sulle registrazioni video effettuate in officina meccanica.

4.3.1 Conversione Standalone e Ispezione con `bvhtocsv`

Nei casi in cui non si desideri attivare il retargeting automatico diretto all'interno di GEM-X, il framework mette a disposizione lo strumento standalone di conversione `bvh_to_csv_converter.py`. Questo modulo riceve in ingresso il file `.bvh` estratto da GEM-X, esegue il retargeting cinematico e genera il dataset tabulare `.csv` per l'Unitree G1.

L'avvio dello strumento avviene all'interno dell'ambiente Conda `gemx`, invocando da terminale lo script con il file di configurazione JSON e abilitando il visualizzatore OpenGL (`--viewer gl`), come mostrato in Figura 4.5.

```
(gemx) C:\Users\Alessandro\GEM-X>python "C:\Users\Alessandro\GEM-X\third_party\soma-retargeter\app\bvh_to_csv_converter.
py" --config "C:\Users\Alessandro\GEM-X\third_party\soma-retargeter\assets\default_bvh_to_csv_converter_config.json" --v
iewer gl
```

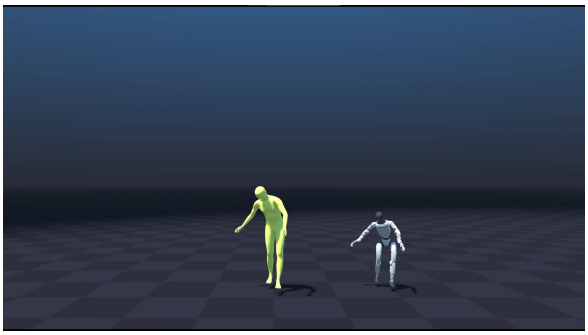
Figura 4.5: Comando CLI per l'esecuzione standalone del convertitore `bvh_to_csv_converter.py` con visualizzatore 3D OpenGL interattivo.

All'avvio, il modulo apre un'interfaccia grafica tridimensionale interattiva (Figura 4.6) che consente all'operatore di selezionare diverse modalità di visualizzazione lungo la sequenza temporale:

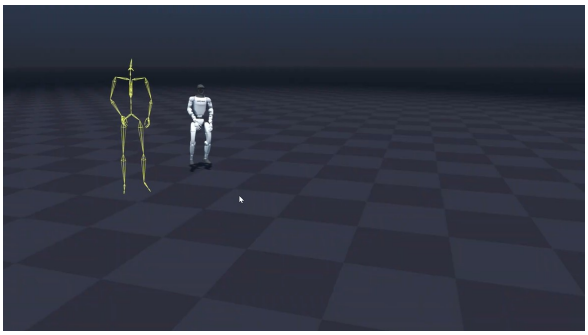
1. **Visualizzazione congiunta affiancata** (Figura 4.6a): mostra simultaneamente lo scheletro umano SOMA e il modello 3D del robot Unitree G1 durante l'esecuzione del gesto, facilitando la verifica immediata della sincronizzazione temporale e dell'assenza di compenetrazioni con il busto;

2. **Dettaglio scheletro umano SOMA** (Figura 4.6b): permette di isolare ed esaminare le rotazioni articolari dei 77 keypoint umani estratti dal video;

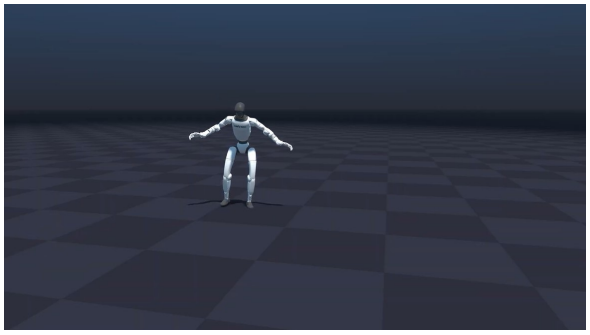
3. **Isolamento del robot Unitree G1** (Figura 4.6c): consente di verificare in modo esclusivo la cinematica dei 29 gradi di libertà del robot e la saldezza dell'appoggio dei piedi al suolo.



(a) Panoramica congiunta scheletro SOMA e robot G1.



(b) Ispezione dello scheletro umano SOMA.



(c) Visualizzazione del solo robot Unitree G1.

Figura 4.6: Interfaccia grafica 3D del modulo `bvhtocsv`: (a) visualizzazione affiancata per il controllo cinematico congiunto, (b) dettaglio dello scheletro SOMA estratto e (c) isolamento del moto articolare del robot Unitree G1.

Riflessione Metodologica sulla Chiusura del Ciclo e Reinforcement Learning. Durante l'ispezione visiva di sequenze cicliche (in cui l'operatore parte da una posa di riposo, esegue una manipolazione e ritorna nella posizione iniziale), si è osservato che il moto ricalcolato dal retargeter non compie una perfetta chiusura uno a uno, presentando un lieve scostamento tra la posa iniziale e quella finale.

Ingegneristicamente, tale discrepanza cinematica non costituisce un limite applicativo: nel paradigma del Reinforcement Learning (che verrà approfondito nel Capitolo 5), le serie temporali estratte da video fungono da solido riferimento primario (motion prior) e guida di comportamento, senza richiedere un inseguimento rigido e vincolante punto per punto. La policy di controllo dinamico a basso livello ha infatti il compito intrinseco di correggere e stabilizzare autonomamente la traiettoria a circuito chiuso, gestendo le forze di contatto al suolo e compensando eventuali disallineamenti della cinematica visiva.

4.3.2 Prove con Variazione della Prospettiva e Registrazioni a Mano Libera

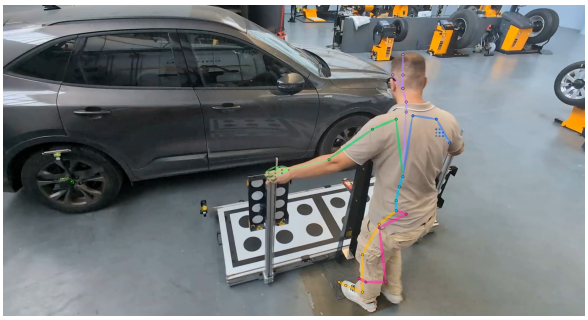
La robustezza geometrica della pipeline è stata testata acquisendo sequenze da molteplici punti di vista: riprese frontali, laterali a 90° e prospettiche a tre quarti (45°). I test

hanno confermato che il modulo Temporal Transformer di GEM-X stima correttamente la profondità e i volumi tridimensionali anche durante riprese effettuate a mano libera con smartphone non fissato su cavalletto (Figura 4.7a, b), gestendo agevolmente i lievi movimenti di brandeggio dell'operatore che registra.

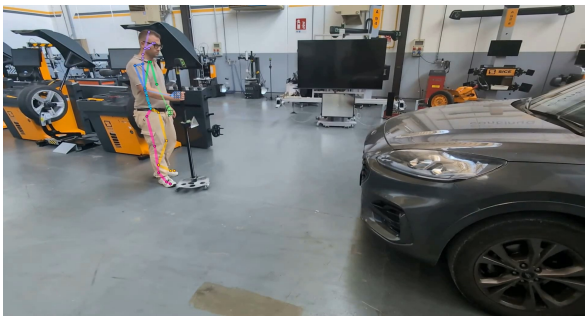
4.3.3 Prove di Manipolazione con Utensili e Confronto di Risoluzione

Un banco di prova particolarmente severo ha riguardato le lavorazioni reali in cui l'operatore maneggia oggetti ingombranti (ruote, pneumatici) e attrezzi metallici (chiavi di serraggio, avvitatori), provocando frequenti occlusioni parziali del corpo e delle mani:

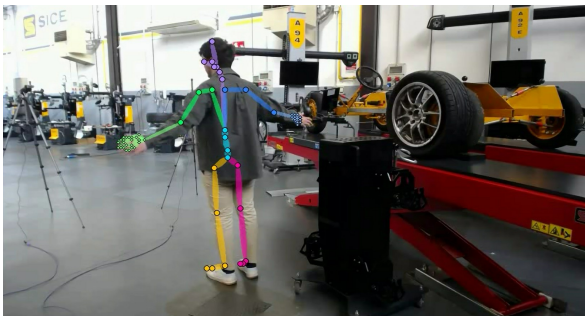
- **Riprese Full HD da Smartphone** (Figura 4.7a, b): registrate a 1920×1080 a 30 fps. Nonostante ampie porzioni degli arti e del busto risultassero coperte dai componenti meccanici durante la posa a terra, il modello ViTPose Whole-Body ha mantenuto tracciati con continuità tutti i keypoint articolari, inferendo correttamente la postura degli arti nascosti;
- **Test Parallelo su Video FreeMoCap a Bassa Risoluzione** (Figura 4.7c, d): per verificare l'effettivo salto qualitativo rispetto alla prima fase sperimentale, la pipeline GEM-X è stata applicata allo stesso identico filmato registrato dalle telecamere di FreeMoCap alla bassa risoluzione forzata di 640×480 pixel. Mentre in FreeMoCap la combinazione MediaPipe + DLT collassava provocando allucinazioni sullo sfondo e scambi di identità (Sezione 2.3), GEM-X ha estratto uno scheletro 3D stabile, coerente e privo di scatti, confermando una netta superiorità anche in condizioni di scarsa definizione ottica.



(a) Ripresa smartphone 1080p con occlusione parziale.



(b) Ripresa smartphone 1080p a mano libera.



(c) Stesso video FreeMoCap a bassa risoluzione (640×480).



(d) Tracking continuo su filmato FreeMoCap degradato.

Figura 4.7: Robustezza del tracking GEM-X sotto occlusione in riprese Full HD da smartphone (a, b) e in sequenze a bassa risoluzione da FreeMoCap (c, d).

4.3.4 Prove su Posture Chine e Ambiente Reale di Officina

Infine, le prove sulle lavorazioni al suolo hanno confermato che la flessione dell'operatore viene gestita correttamente senza sprofondamenti virtuali: il modulo di stabilizzazione blocca i piedi a quota $Z = 0$ e la cinematica del baricentro evolve fluidamente. Inoltre, in presenza di illuminazione mista, ombre dei ponti sollevatori e tute scure, il tracker YOLOX + ByteTrack ha mantenuto l'aggancio continuativo senza falsi allarmi sullo sfondo.

4.3.5 Sintesi dei Risultati e Ponte verso la Simulazione Dinamica

L'esito complessivo della campagna sperimentale è riepilogato nella Tabella 4.1, che evidenzia il successo della pipeline nel convertire video reali in traiettorie articolari pronte per il simulatore fisico.

Scenario di Prova	Condizione Operativa	Risultato Sperimentale
Ispezione Cinematica	Sequenze complete di lavoro	Movimento fluido e congruente, zero compenetrazioni
Riprese Smartphone	Full HD (1080p) a mano libera	Posa 3D stabile anche con lievi vibrazioni della camera
Occlusioni con Utensili	Attrezzi e componenti ruota	Tracking continuo di mani e dita senza perdita di punti
Confronto Low-Res	Stesso video FreeMoCap (640×480)	Nessun collasso né allucinazione rispetto a MediaPipe (FreeMoCap)
Lavorazioni Chine	Flessioni e piegamenti a terra	Piedi saldi al suolo ed eliminazione del foot skating
Ambiente Industriale	Luci d'officina e ombre	Tracciamento continuo senza perdite del soggetto

Tabella 4.1: Quadro di sintesi delle prove sperimentali di validazione della pipeline GEM-X + SOMA Retargeter.

Ponte Metodologico verso il Capitolo 5: Convalidate sperimentalmente la solidità, la fedeltà geometrica e la continuità delle traiettorie cinematiche generate per i 29 motori del robot Unitree G1, il successivo Capitolo 5 affronterà la validazione del comportamento in ambiente fisico simulato (inclusa la dinamica multibody, la gravità, i carichi inerziali e le forze di contatto al suolo) all'interno del framework NVIDIA Isaac Lab.

Capitolo 5

Proof of Concept in Simulazione Dinamica (Isaac Lab) e Analisi Critica dell'Imitazione

Questo capitolo presenta la validazione in simulazione dinamica tramite NVIDIA Isaac Lab, impiegato come Proof of Concept (PoC) per testare l'inseguimento delle traiettorie estratte da video sull'Unitree G1. Vengono descritti l'ecosistema di simulazione, l'integrazione plug-and-play del parser CSV e un'analisi critica dei limiti biomeccanici legati all'imitazione umana pura.

5.1 Introduzione agli Ambienti di Simulazione: Isaac Sim e Isaac Lab

La validazione dinamica di traiettorie cinematiche su robot umanoidi complessi richiede piattaforme di simulazione capaci di modellare con elevata accuratezza contatti rigidi, attriti non lineari, dinamiche attuatorie e risposte sensoriali. In questo contesto, l'ecosistema NVIDIA si è affermato come il nuovo paradigma di riferimento per la Physical AI.

5.1.1 Panoramica: Motore Fisico vs Framework RL

È fondamentale operare una netta distinzione concettuale e architetturale tra i due livelli dell'ambiente NVIDIA adottato, illustrati schematicamente in Figura 5.1:

- **NVIDIA Isaac Sim** [6]: rappresenta il motore di simulazione fisica, fotorealistica e di rendering. Basato sulla piattaforma NVIDIA Omniverse e sul formato standard Pixar OpenUSD [25], integra il solutore fisico avanzato PhysX 5, gestendo il rilevamento accurato delle collisioni, la deformazione dei contatti e la simulazione realistica

dei sensori fisici (camere RGB-D, LiDAR e IMU) tramite ray-tracing accelerato da GPU.

- **NVIDIA Isaac Lab** [18]: costituisce il framework applicativo open-source modulare, sviluppato come evoluzione di IsaacGymEnvs e Orbit, esplicitamente progettato per il Robot Learning e il Reinforcement Learning (RL). Isaac Lab astrae l'interfaccia verso il simulatore fornendo primitive di alto livello per la definizione di compiti MDP (Markov Decision Process), logiche di reward, randomizzazione degli ambienti e vettorizzazione parallela su GPU.

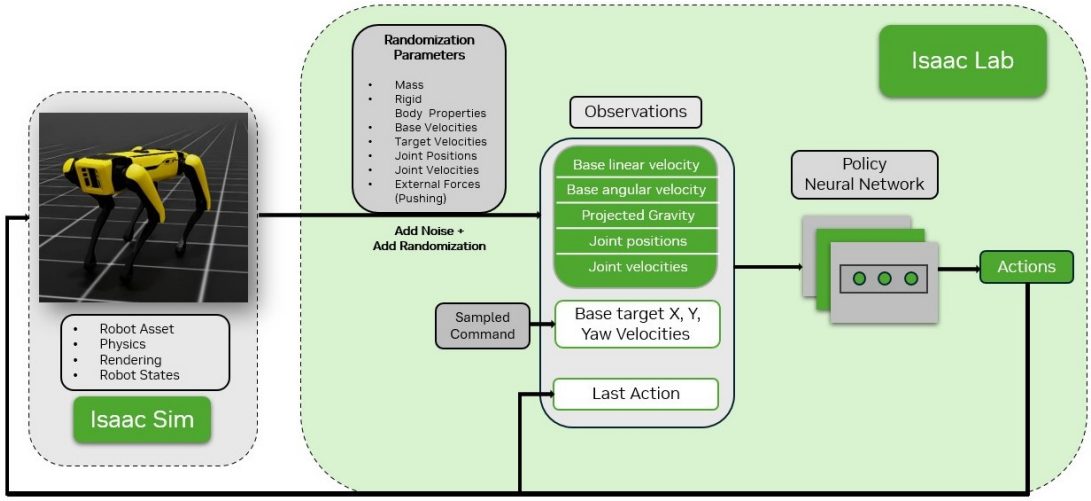


Figura 5.1: Suddivisione dei livelli e ambiti operativi tra Isaac Sim (motore fisico e rendering) e Isaac Lab (framework di robot learning) (fonte: [19]).

5.1.2 Vantaggi sui Competitor e Riduzione del Sim-to-Real Gap

Rispetto ai simulatori tradizionali largamente impiegati nella comunità accademica e industriale, quali MuJoCo [31], PyBullet o Gazebo, la suite Isaac Sim/Lab introduce un salto generazionale fondato sulla computazione tensorizzata end-to-end su GPU.

Nei motori fisici storici, l'avanzamento dello stato di simulazione avviene serialmente sulla CPU, introducendo pesanti colli di bottiglia nel trasferimento dati via bus PCIe verso la GPU per l'addestramento delle reti neurali. In Isaac Lab, l'intero ciclo di vita dell'ambiente (calcolo della cinematica e della dinamica, rilevamento dei contatti, generazione delle osservazioni e aggiornamento dei pesi di policy) risiede permanentemente nella VRAM della GPU tramite tensori PyTorch [24]. Ciò consente di simulare contemporaneamente da centinaia a decine di migliaia di istanze del robot con frequenze di campionamento globali superiori a milioni di passi al secondo.

Inoltre, la suite NVIDIA nasce con l'obiettivo primario di abbattere il divario di realtà (Sim-to-Real gap): la fedeltà del solutore PhysX 5 e l'integrazione di pipeline fotorealistiche

RTX permettono di trasferire direttamente le policy addestrate su hardware reale senza richiedere ricalibrazioni manuali estenuanti, come illustrato dal paradigma operativo in Figura 5.2.

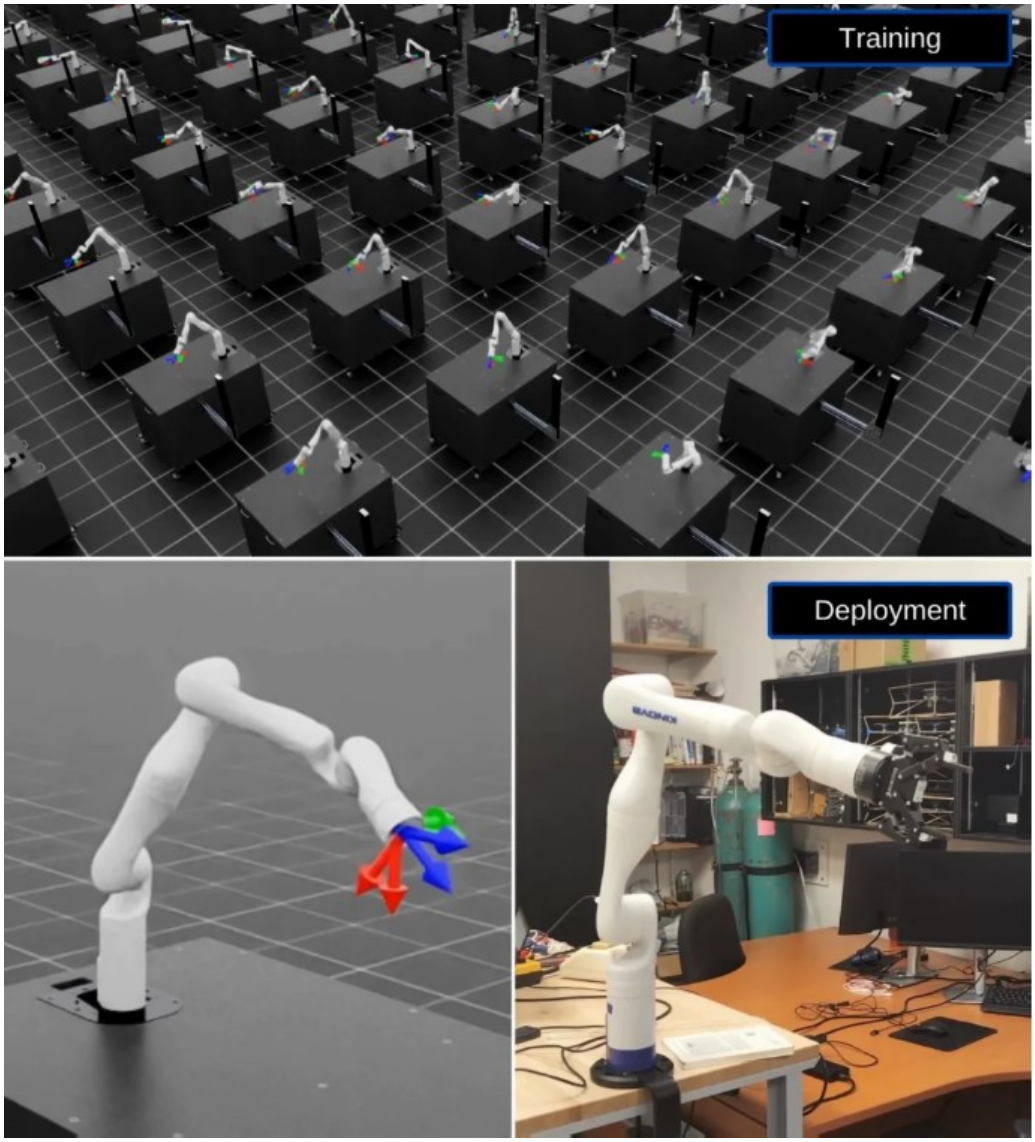


Figura 5.2: Rappresentazione del paradigma Sim-to-Real: addestramento massivo parallelo in simulatore GPU (Training) e successivo trasferimento della policy sul braccio robotico (fonte: [13]).

5.1.3 Le Sfide dell'Adozione: Strumento all'Avanguardia ma Acerbo

Nonostante l'enorme potenziale computazionale, l'adozione di Isaac Sim e Isaac Lab comporta sfide ingegneristiche e vincoli infrastrutturali non trascurabili:

1. **Requisiti hardware elevati**: l'esecuzione fluida dell'ambiente richiede workstation dotate di schede video NVIDIA dedicate ad alte prestazioni con abbondante memoria VRAM, precludendo l'utilizzo su computer convenzionali o con GPU integrate;

2. **Documentazione in rapida evoluzione**: trattandosi di un framework giovane e in continuo sviluppo, la documentazione ufficiale presenta talvolta aree frammentate o non perfettamente allineate agli ultimi aggiornamenti di versione;

3. **Stabilità delle release**: il ciclo frequente di rilasci introduce occasionali modifiche di compatibilità (breaking changes) nelle API di configurazione;

4. **Curva di apprendimento ripida**: richiede una solida padronanza congiunta di programmazione asincrona in Python, ecosistema USD/Omniverse, calcolo parallelo in CUDA e logiche avanzate di Reinforcement Learning vettorizzato.

5.2 Motivazioni e Scopo del Proof of Concept (PoC)

5.2.1 La Scelta di Strumenti Open-Source State-of-the-Art

La decisione di integrare Isaac Lab nella presente attività di tesi deriva dalla precisa volontà di confrontarsi con gli strumenti state-of-the-art (SOTA) più evoluti a livello internazionale. Nell'ottica di un reparto R&D all'avanguardia come quello di Nexion Group, sperimentare direttamente le tecnologie emergenti consente di valutare in anticipo la scalabilità futura dei flussi di lavoro, acquisendo competenze dirette sulla frontiera della robotica umanoide.

5.2.2 Perimetro di Utilizzo: Ricerca e Rimodulazione di Progetti Esistenti

È doveroso chiarire il perimetro applicativo del lavoro svolto: l'obiettivo non è stato quello di sviluppare un framework di Reinforcement Learning interamente da zero, operazione che richiederebbe mesi di lavoro dedicati unicamente all'infrastruttura di basso livello.

Lo scopo primario è consistito nella validazione end-to-end (Proof of Concept, PoC) dell'intera pipeline: dimostrare cioè che le serie temporali cinematiche estratte da un semplice video monoculare (Capitolo 4) possono essere ingerite, tracciate e stabilizzate dinamicamente da un modello robotico realistico.

Per raggiungere tale traguardo in modo pragmatico ed efficace, si è resa necessaria un'approfondita fase preliminare di indagine e studio attraverso la vasta, frammentata e complessa documentazione open-source di Isaac Lab, volta a individuare una base di lavoro solida e compatibile. Tale ricerca ha consentito di scoprire l'esistenza dell'ambiente di riferimento per la locomozione del robot Unitree G1 (denominato G1 Velocity / G1 Rough). Partendo da questo modello di base, ne sono stati rimodellati radicalmente i

generatori di comandi e le funzioni di ricompensa, convertendo un semplice task di marcia a velocità fissa in un'avanzata architettura di inseguimento posturale e cinematico.

5.3 Dinamiche di Apprendimento: RL Massivo e Domain Randomization

5.3.1 Parallelizzazione su GPU e Algoritmo PPO

L'addestramento della policy di controllo si basa sull'algoritmo Proximal Policy Optimization (PPO) [29], implementato tramite la libreria ad alte prestazioni `rsl_rl` [27]. La parallelizzazione massiva consente di istanziare $N = 4096$ ambienti simultanei all'interno di una singola GPU.

In ogni passo temporale, la policy neurale (architettura Actor-Critic basata su Multi-Layer Perceptron) riceve le osservazioni da tutti i 4096 robot in parallelo, calcola le azioni per i 29 giunti e riceve le relative reward, accumulando milioni di esperienze al minuto e riducendo a poche decine di minuti tempi di addestramento che su architetture CPU-based richiederebbero giorni interi.

5.3.2 Domain Randomization come Ponte verso la Realtà

Uno dei punti cardine per garantire la robustezza del controllo dinamico è la Domain Randomization (DR). Nel simulatore, i parametri fisici dell'ambiente non sono fissi ma vengono perturbati stocasticamente a ogni reset degli episodi:

- **Randomizzazione della posizione di spawn e orientamento iniziale**: a ogni reset dell'episodio, il robot non viene generato sempre nella medesima posa e coordinata fissa, ma subisce piccole traslazioni e rotazioni casuali sul piano del terreno. Questa variabilità iniziale impedisce alla rete di memorizzare traiettorie predefinite e la costringe a esplorare attivamente uno spazio di stati molto più ampio, favorendo l'emergenza di strategie di bilanciamento flessibili e indipendenti dal punto di partenza;
- **Variazione dell'attrito**: modulazione del coefficiente di attrito statico e dinamico del suolo (da superfici scivolose ad alto grip);
- **Masse e carichi stocastici**: aggiunta di masse stocastiche aggiuntive sulla base e sui link del robot, simulando variazioni di carico o tolleranze costruttive;
- **Perturbazioni esterne (Push Recovery)**: applicazione di forze impulsive casuali per stimolare la capacità di recupero dell'equilibrio;

- **Rumore e latenze sensoriali**: randomizzazione dei ritardi di attuazione (actuator latency) e iniezione di rumore gaussiano sulle letture sensoriali (encoder articolari e IMU).

Sebbene il presente Proof of Concept si concentri sulla validazione in simulazione, l'adozione della Domain Randomization garantisce che la policy non sviluppi comportamenti sovra-adattati (overfitting) alle condizioni ideali del simulatore, preparando la base algoritmica per il futuro deployment su hardware fisico.

5.4 Configurazione Pratica del Task per l'Unitree G1

5.4.1 Asset USD, Inerzie e Collisioni

La definizione del robot all'interno del simulatore avviene mediante l'asset ufficiale in formato Universal Scene Description (`.usd`) fornito per l'Unitree G1 [32]. Il file incorpora l'esatta gerarchia cinematica dei 29 gradi di libertà, le matrici d'inerzia calcolate sui modelli CAD, i limiti fisici di escursione angolare e le mesh di collisione approssimate (cilindri e capsule convesse) necessarie a consentire un calcolo ad altissima velocità dei punti di contatto senza sovraccaricare il solutore PhysX.

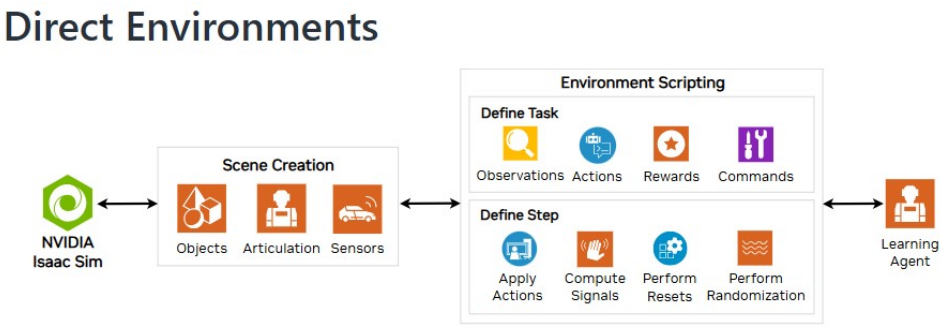
5.4.2 Paradigmi Architetturali: Manager-Based vs Direct Workflow

Isaac Lab offre due differenti paradigmi di sviluppo per la definizione dei compiti di apprendimento [20], illustrati in Figura 5.3:

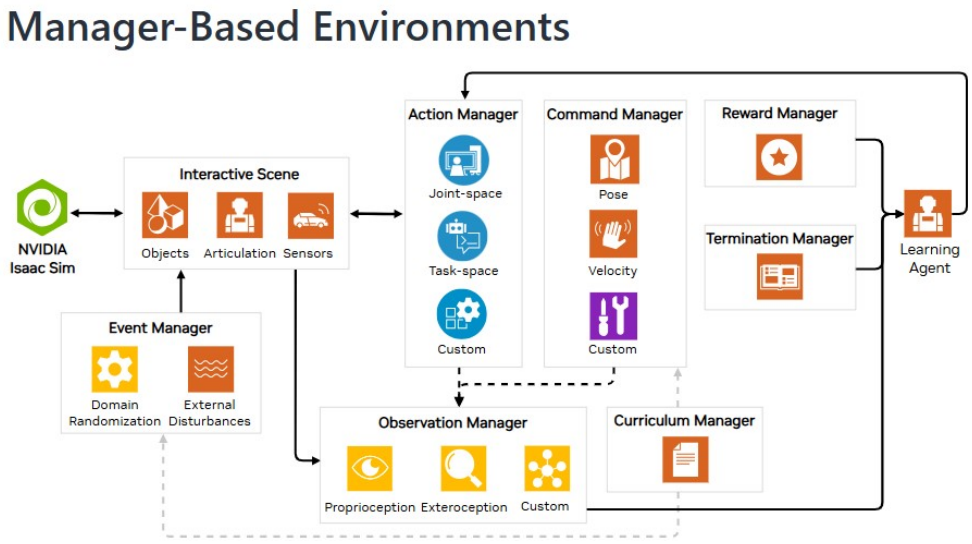
1. **Direct Workflow** (Figura 5.3a): approccio a basso livello in cui l'intero ambiente di simulazione viene programmato all'interno di una singola classe monolitica che opera direttamente sui tensori CUDA. Sebbene consenta di massimizzare il throughput computazionale riducendo al minimo l'overhead di astrazione, sacrifica la leggibilità del codice e rende onerosa la manutenzione o l'estensione a nuovi sensori;

2. **Manager-Based Workflow** (Figura 5.3b): architettura a oggetti modulare e altamente strutturata. L'ambiente è composto da manager indipendenti (Observation Manager, Action Manager, Reward Manager, Command Manager, Event Manager, Termination Manager), ciascuno configurato tramite classi dichiarative.

Per lo sviluppo del presente Proof of Concept è stato adottato l'approccio Manager-Based: tale scelta ha garantito un'eccellente pulizia ingegneristica, consentendo di sostituire agilmente il solo generatore di comandi (Command Manager) e le relative funzioni di ricompensa (Reward Manager) senza dover alterare la complessa infrastruttura meccatronica del robot o il ciclo di reset.



(a) Architettura Direct Workflow: struttura monolitica orientata al massimo throughput tensorizzato.



(b) Architettura Manager-Based Workflow: decomposizione modulare in manager specializzati.

Figura 5.3: Confronto tra i due paradigmi architetturali di NVIDIA Isaac Lab per la creazione di ambienti di addestramento (fonte: [20]).

5.4.3 Spazio delle Osservazioni, Azioni e Reward Engineering

L'apprendimento per rinforzo modella l'interazione tra robot e ambiente attraverso tre elementi cardine (formalizzati nel processo MDP):

- **Spazio delle Osservazioni ($\mathcal{S}$, input della rete)**: rappresenta l'insieme delle informazioni fornite alla rete a ogni passo temporale, articolato in:

  1. Stato attuale del robot (percezione propriocettiva): posizioni e velocità istantanee dei 29 motori, unitamente all'inclinazione e alle velocità angolari del busto misurate dall'IMU;
  2. Obiettivo richiesto (comando da inseguire): posizioni angolari target e velocità desiderate per il centro di massa lette dal fotogramma corrente del file CSV.

- **Spazio delle Azioni ($\mathcal{A}$, output della rete)**: vettore di 29 valori continui generato dalla policy neurale, rappresentante gli offset angolari desiderati per i giunti; tali target di posizione vengono convertiti istantaneamente in coppie motrici/correnti (τ) tramite regolatori locali PD (Proporzionale-Derivativo) integrati su ciascun attuatore.

- **Reward Engineering (Funzione di Ricompensa)**: segnale scalare globale che aggrega la fedeltà di tracciamento di tutti i 29 giunti e della velocità della base, introducendo penalità per cadute, movimenti a scatto (jerk) ed eccessivo consumo energetico.

5.5 Sviluppo del Parser CSV e Integrazione in Isaac Lab

Il nucleo pratico del Proof of Concept (PoC) risiede nella capacità di interfacciare i dati cinematici estratti dalla pipeline visiva (Capitolo 4) con il motore di Reinforcement Learning. All'interno del paradigma Manager-Based, ciò ha richiesto lo sviluppo di due componenti custom integrati nell'ecosistema in Python: un Command Generator per il parsing dei file CSV e una serie di funzioni di ricompensa per l'inseguimento del riferimento.

5.5.1 Generatore di Comandi e Mappatura Cinematica

Lo script `csv_command_generator.py` è stato ingegnerizzato per sostituire i tradizionali comandi manuali dell'utente (come joystick o tastiera) con le serie temporali cinematiche estratte dal file CSV.

In primo luogo, il modulo esegue la mappatura cinematica dei dati: le colonne generate da SOMA Retargeter (ad esempio `left_hip_pitch_joint_dof`) vengono collegate

dinamicamente ai 29 giunti previsti dal modello dell'Unitree G1 all'interno di Isaac Lab, occupandosi contestualmente della conversione delle ampiezze angolari da gradi a radianti.

In secondo luogo, lo script adatta il movimento registrato trasformandolo dal punto di vista globale della stanza al punto di vista locale del robot. Nel video originale, la persona si sposta nello spazio secondo coordinate assolute fisse; tuttavia, per permettere alla rete neurale di imparare a camminare in qualsiasi direzione e scenario, le velocità di avanzamento, spostamento laterale e rotazione devono essere riferite al corpo stesso del robot. In questo modo, l'indicazione di avanzare o curvare fa sempre riferimento alla parte frontale del busto dell'umanoide, rendendo il controllo indipendente dalla posizione o dall'orientamento iniziale nella scena.

5.5.2 Decorrelazione Spaziotemporale (Resampling) per il Reinforcement Learning

Addestrando in parallelo migliaia di agenti robotici simultaneamente su GPU tramite l'algoritmo PPO (Proximal Policy Optimization), far iniziare tutti gli ambienti allo stesso istante ($t = 0$ del file CSV) produrrebbe gradienti fortemente correlati, destabilizzando la convergenza della rete neurale Actor-Critic e impedendo un'esplorazione uniforme dello spazio degli stati.

Per risolvere questo problema, nel Command Generator è stata implementata una logica di resampling spaziotemporale: a ogni reset dell'ambiente, l'indice di partenza del video per quello specifico robot (`env_frame`) viene inizializzato con un numero intero casuale estratto tra 0 e la lunghezza totale del CSV. In questo modo, l'agente esplora uniformemente l'inizio, il centro e la fine del movimento fin dalle prime epoche di addestramento.

5.5.3 Funzioni di Reward basate su Kernel Esponenziale e Termini di Fluidità

L'apprendimento per imitazione cinematica (Motion Mimicry) si concretizza attraverso le funzioni di costo definite nello script `csv_rewards.py`. Il principio matematico di base calcola la distanza quadratica euclidea tra lo stato del robot e il target, ossia errore $= \sum(q_i - q_i^*)^2$: tale formulazione ha il vantaggio intuitivo di tollerare piccole imperfezioni fisiologiche (per errori piccoli l'elevamento al quadrato genera penalità minime) punendo invece severamente le grandi discrepanze posturali.

Tuttavia, se utilizzato direttamente come ricompensa lineare o quadratica pura ($-\sum e_i^2$), nelle fasi iniziali dell'addestramento, quando la policy è casuale e il robot si trova molto distante dal moto desiderato, l'errore totale assumerebbe valori negativi elevatissimi. Nel Reinforcement Learning, tali picchi causano il fenomeno dei gradienti esplosivi, destabilizzando la convergenza dell'algoritmo PPO.

Per risolvere questo problema, l'errore quadratico viene inserito all'interno di un Kernel Esponenziale:

$$R = \exp\left(-\frac{\text{errore}}{\sigma}\right) \tag{5.1}$$

dove σ rappresenta un parametro di scala (temperatura). Questo artificio matematico garantisce che il segnale di premio sia rigorosamente limitato nell'intervallo $[0, 1]$:

- A errore nullo ($q = q^*$), il premio è massimo ($R = 1.0$);
- Per errori contenuti, la ricompensa decresce in modo continuo e differenziabile;
- Per errori macroscopici o cadute, il premio tende asintoticamente a zero ($R \to 0$), evitando che la funzione di costo diverga negativamente a $-\infty$ e preservando la stabilità numerica del gradiente.

Sfruttando tale formulazione, il sistema di reward complessivo si articola su tre componenti:

- **Inseguimento Posturale (`track_joint_pos_csv`)**: applica il Kernel Esponenziale sulla somma degli errori quadratici dei 29 angoli target estratti dal CSV, vincolando il robot ad assumere la sagoma umana richiesta.
- **Inseguimento del Centro di Massa (`track_base_vel_csv`)**: verifica che il robot stia traslando nello spazio coerentemente con le velocità lineari e angolari calcolate dall'odometria (v_x, v_y, ω_z). Funge da regolarizzatore critico contro il fenomeno dello scivolamento da fermo (foot skating), penalizzando un robot che si limiti a muovere le gambe sul posto senza produrre reale avanzamento.
- **Regolarizzazione della Fluidità e Velocità (Smoothness Penalties)**: per garantire un controllo armonico ed eliminare oscillazioni parassite, l'ambiente applica due termini di costo complementari:
  - Penalità sulle accelerazioni articolari (`dof_acc_l2`): scoraggia variazioni brusche di velocità sui singoli giunti, prevenendo vibrazioni e tremolii ad alta frequenza;
  - Penalità sul tasso di variazione dell'azione (`action_rate_l2`): penalizza la differenza quadratica tra comandi consecutivi $(\mathbf{a}_t - \mathbf{a}_{t-1})^2$, forzando la rete a generare profili di attuazione continui ed eliminando dannosi colpi d'ariete sui riduttori del G1.

5.5.4 Il rischio di Reward Hacking, Tuning dei Pesi e Validazione Visiva

Un aspetto metodologico fondamentale nell'apprendimento per rinforzo risiede nella calibrazione empirica dei coefficienti di peso (w_i) associati a ciascuna funzione di costo.

La policy neurale è un ottimizzatore matematico puro: se le funzioni obiettivo non sono perfettamente bilanciate, la rete tenderà a sfruttare scappatoie analitiche per massimizzare il punteggio numerico complessivo a discapito della reale correttezza fisica del moto, manifestando il noto fenomeno del Reward Hacking (o Specification Gaming).

Un esempio emblematico documentato nella letteratura e nella divulgazione sperimentale (come negli esperimenti di locomozione su robot quadrupedi tipo Ant [2], illustrato in Figura 5.4) vede l'agente imparare ad avanzare sollevando costantemente una zampa in aria e muovendosi su tre arti: tale postura aberrante consente alla rete neurale di massimizzare la ricompensa di velocità istantanea minimizzando al contempo le penalità di collisione, attrito o consumo energetico dell'arto sollevato.

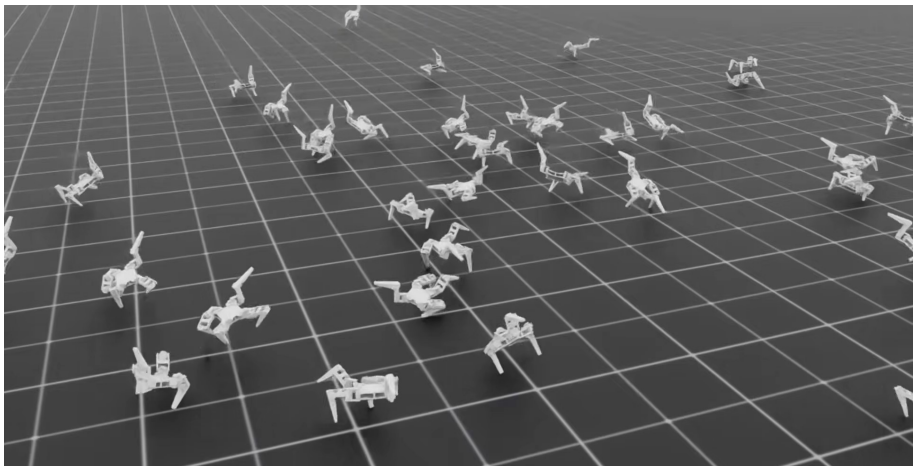


Figura 5.4: Esempio di Reward Hacking durante l'addestramento RL di un robot quadrupede tipo Ant: la policy massimizza la ricompensa trascinandosi su tre zampe ed evitando contatti con il quarto arto (fonte: [2]).

Nel contesto del presente lavoro sull'Unitree G1, il rischio principale era che la policy trovasse il modo di "imbrogliare" rimanendo comodamente ferma sul posto: limitandosi a muovere braccia e gambe in aria per massimizzare il premio di posa (`track_joint_pos_csv`), il robot avrebbe evitato qualsiasi rischio di instabilità o caduta connesso all'avanzamento. L'introduzione congiunta dell'inseguimento vincolante della velocità del baricentro (`track_base_vel_csv`) e un accurato tuning dei pesi hanno impedito questa scappatoia, imponendo un reale trasporto del centro di massa nello spazio tridimensionale.

Da ciò discende una linea guida operativa imprescindibile: l'analisi delle metriche scalari su TensorBoard deve essere costantemente affiancata da sessioni periodiche di validazione visiva nel visualizzatore 3D di Isaac Sim. Solo l'ispezione diretta dell'animazione fisica permette all'ingegnere di rilevare comportamenti parassiti o asimmetrie meccatroniche, guidando la regolazione fine dei pesi per evitare che i singoli termini entrino in conflitto distruttivo tra loro.

5.6 Esecuzione dell'Addestramento e Validazione del Proof of Concept

5.6.1 Configurazione Plug-and-Play e Lancio su Terreni Paralleli

Un punto di forza architetturale dell'approccio implementato è la natura rigorosamente plug-and-play del flusso dati: per addestrare l'agente su una nuova sessione di movimento catturata dal video monoculare, l'operatore non deve modificare alcuna riga di codice dell'infrastruttura di simulazione o della rete neurale. È sufficiente specificare il percorso del file esportato da SOMA Retargeter e la frequenza di campionamento nel file di configurazione dell'ambiente (Lista 5.1):

```
self.commands.base_velocity = CsvVelocityCommandCfg(
    csv_path=r"percorso/verso/traiettoria_retarget_g1.csv",
    dt=1.0 / 30.0, # Sincronizzazione con il framerate del video
    (30 FPS)
    resampling_time_range=(10000.0, 10000.0), # Disattiva il
   resampling casuale durante l'episodio
)
```

Lista 5.1: Istanziazione dichiarativa del comando CSV nella configurazione dell'ambiente.

Grazie al meccanismo di ereditarietà ad oggetti di Isaac Lab, qualsiasi configurazione derivata (es. `flat_env_cfg.py` per terreno pianeggiante o `rough_env_cfg.py` per suolo irregolare con dislivelli) acquisisce automaticamente il riferimento cinematico.

L'esecuzione dell'addestramento su piattaforma Windows avviene all'interno dell'ambiente Conda `env_isaaclab`, invocando lo script di training di `rsl_rl` con il task impostato su `Isaac-Velocity-Rough-G1-v0` e 4096 istanze parallele, come illustrato dal comando da terminale in Figura 5.5a. L'argomento `-headless` disabilita il rendering grafico interattivo, scaricando completamente la GPU dal carico di visualizzazione e destinando il 100% della banda di calcolo alla simulazione fisica dei contatti e all'inferenza tensoriale.

Durante l'evoluzione del training, il framework provvede a salvare automaticamente a intervalli regolari (ogni N iterazioni configurate nei parametri di `rsl_rl`) un file di checkpoint contenente lo stato completo della rete neurale e dell'ottimizzatore. Questo salvataggio incrementale offre un duplice vantaggio: preserva l'avanzamento contro eventuali interruzioni e permette di caricare in qualsiasi momento uno stadio intermedio per analizzarne visivamente il comportamento.

Una volta completata l'ottimizzazione dei pesi (oppure per ispezionare qualitativamente uno dei checkpoint generati), viene eseguito successivamente lo script di play puntando al task `Isaac-Velocity-Rough-G1-Play-v0` (Figura 5.5b).

Questa seconda fase riattiva l'interfaccia grafica interattiva di Isaac Sim / Omniverse (Figura 5.6), consentendo all'operatore di verificare visivamente la fluidità dei passi, la stabilità dell'equilibrio e l'effettiva aderenza alla traiettoria estratta dal video mocap attraverso una griglia di terreni procedurali eterogenei (pendenze, gradini piramidali, blocchi sconnessi e suolo irregolare).

```
2026-07-30T09:15:50Z [22,257ms] [Warning] [carb] Client gpu.foundation.plugin has acquired [gpu::unstable::IMemoryBudget
ManagerFactory v0.1] 100 times. Consider accessing this interface with carb::getCachedInterface() (Performance warning)
^C
(env_isaaclab) C:\Users\Alessandro\IsaacLab>python scripts\reinforcement_learning\rsl_rl\train.py --task Isaac-Velocity-
Rough-G1-v0 --num_envs 4096 --headless
```

(a) Comando CLI per l'avvio del training massivo su GPU (4096 ambienti in modalità headless).

```
2026-07-30T09:15:50Z [22,257ms] [Warning] [carb] Client gpu.foundation.plugin has acquired [gpu::unstable::IMemoryBudget
ManagerFactory v0.1] 100 times. Consider accessing this interface with carb::getCachedInterface() (Performance warning)
^C
(env_isaaclab) C:\Users\Alessandro\IsaacLab>python scripts\reinforcement_learning\rsl_rl\play.py --task Isaac-Velocity-R
ough-G1-Play-v0
```

(b) Comando CLI per il successivo lancio del play e la visualizzazione interattiva della policy in Isaac Sim.

Figura 5.5: Sequenza operativa da terminale nell'ambiente `env_isaaclab`: (a) fase preliminare di addestramento intensivo e (b) successiva fase di inferenza e validazione visiva del movimento dell'Unitree G1.

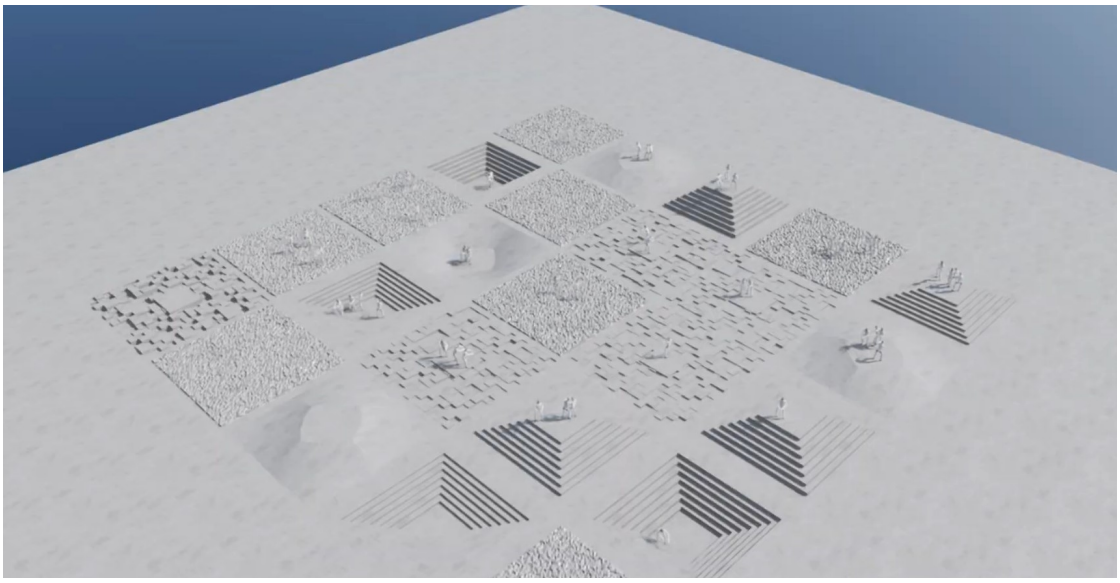
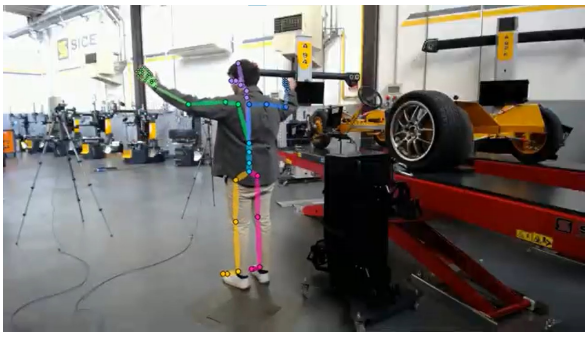


Figura 5.6: Ambiente di simulazione 3D in Isaac Sim durante la validazione visiva (play): istanze dell'Unitree G1 che eseguono la policy di locomozione e posa su una matrice di terreni procedurali complessi.

Per validare in modo immediato la catena end-to-end della pipeline durante la fase di sviluppo, è stato introdotto intenzionalmente nella registrazione video un movimento posturale caratteristico e facilmente riconoscibile (keyframe). Come mostrato nel confronto in Figura 5.7, il robot Unitree G1 riproduce fedelmente in simulazione la posa a braccia aperte del soggetto ripreso estratta dalla stima di posa monoculare, confermando visivamente la qualità del retargeting e l'efficacia dell'apprendimento per imitazione.



(a) Video del soggetto con overlay della stima di posa.



(b) Robot Unitree G1 in simulazione che imita la posa.

Figura 5.7: Validazione qualitativa della pipeline tramite keyframe: confronto tra la posa del soggetto nel video monoculare (a) e la replica del movimento in Isaac Sim al termine del training (b).

5.6.2 Telemetria da Riga di Comando ed Evoluzione dell'Addestramento

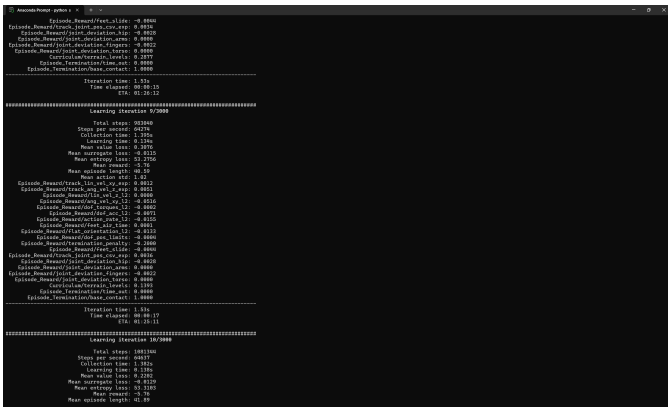
Durante l'esecuzione dell'addestramento in modalità headless, il framework stampa periodicamente a terminale un prospetto analitico con il riepilogo delle metriche per ogni iterazione di apprendimento. In Figura 5.8 viene documentata l'evoluzione temporale completa dell'esperimento condotto sulla traiettoria estratta dal video dell'operatore in officina (la lavorazione con utensile appoggiato sullo pneumatico analizzata nel Capitolo 4):

Dall'ispezione quantitativa dei tre stadi emergono evidenze di grande rilievo ingegneristico:

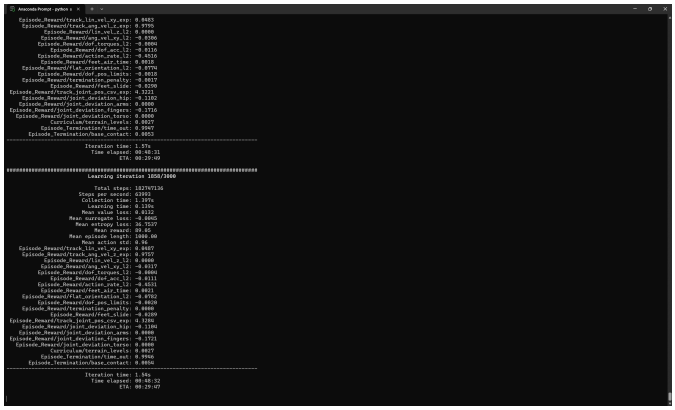
1. **Throughput Computazionale Costante**: il sistema mantiene una frequenza di simulazione estremamente elevata e stabile, compresa tra 63.993 e 67.324 passi al secondo (steps per second), consentendo di simulare quasi 300 milioni di passi fisici in appena 4775.88 secondi ($\sim$ 1 ora e 19 minuti);

2. **Transizione della Reward Media**: il valore della ricompensa globale passa da un valore iniziale negativo di -5.76 (in cui dominano le penalità di caduta e di scatto articolare) a $+89.05$ a metà processo, fino a raggiungere stabilmente il valore di convergenza di $+94.02$;

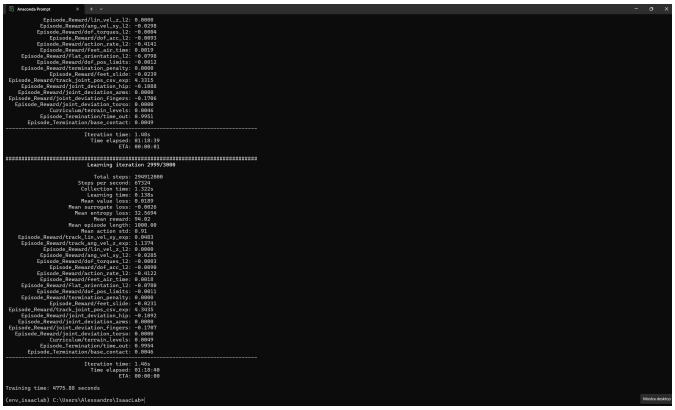
3. **Stabilizzazione dell'Equilibrio Dinamico**: la durata media dell'episodio cresce rapidamente da 41.89 passi iniziali fino a saturare il limite massimo dell'orizzonte impostato (1000.00 passi). Al termine del training, la percentuale di episodi che si concludono per naturale scadenza del tempo (`time_out`) raggiunge il 99.54%, mentre i contatti accidentali del corpo con il suolo (`base_contact`) crollano ad appena lo 0.46%;



(a) Inizio addestramento (iterazione 10/3000): policy iniziale acerba con ricompensa negativa e cadute frequenti.



(b) Fase intermedia (iterazione 1858/3000): rapido miglioramento dell'equilibrio e crescita della ricompensa.



(c) Fine addestramento (iterazione 3000/3000): convergenza completata con movimento fluido e stabile.

Figura 5.8: Monitoraggio da terminale CLI delle tre fasi dell'addestramento PPO in Isaac Lab (traiettoria dell'operatore con utensile su pneumatico): (a) avvio iniziale, (b) consolidamento intermedio e (c) convergenza a regime.

4. **Accuratezza del Tracking Posturale**: la ricompensa esponenziale associata all'inseguimento della postura dell'operatore (`track_joint_pos_csv_exp`) compie un balzo da 0.0036 a 4.3435, a fronte di un massimo teorico di 5.0 (weight (5) $\times$ reward exp([0,1])), a dimostrazione che il modello ha appreso a guidare fedelmente le 29 articolazioni del robot G1 secondo il gesto registrato nel video.

5.6.3 Monitoraggio Grafico e Telemetria Avanzata (TensorBoard)

In parallelo ai log testuali da riga di comando, l'operatore può visualizzare in tempo reale l'evoluzione analitica delle curve di convergenza tramite la dashboard web interattiva di TensorBoard, attivata con il comando documentato in Figura 5.9a ed esposta su `http://localhost:6006/`.

Attraverso la dashboard grafica (Figura 5.9b), il sistema registra l'evoluzione analitica di tutti i parametri del processo di ottimizzazione:

1. **Loss di Policy e Value Function**: progressiva convergenza delle due reti dell'architettura Actor-Critic. L'errore del Critico si riduce drasticamente man mano che impara a stimare il valore delle ricompense future, mentre la Policy dell'Actor si stabilizza passando da un'iniziale esplorazione casuale a una strategia di camminata sicura, fluida ed esperta;

2. **Tracking Reward Posturale (`track_joint_pos_csv`)**: incremento rapido della ricompensa nelle prime 10^6 interazioni, indicativo di un rapido allineamento delle articolazioni superiori e inferiori agli angoli del riferimento mocap;

3. **Tracking Reward della Base (`track_base_vel_csv`)**: apprendimento progressivo della spinta al suolo necessaria a trasferire il centro di massa in avanti alla velocità desiderata, prevenendo oscillazioni parassite da fermo;

4. **Regolarizzazione e Penalità (`action_rate_l2`, `dof_acc_l2`)**: progressivo smorzamento delle accelerazioni improvvise e stabilizzazione delle coppie erogate, a testimonianza di un controllo motorio progressivamente più fluido e privo di scatti.

La validazione qualitativa e quantitativa ha dimostrato che il robot Unitree G1 è in grado di mantenere stabilmente l'equilibrio dinamico su terreni piani e moderatamente sconnessi, riproducendo con fedeltà le traiettorie dell'operatore umano estratte dal video monoculare. Ciò sancisce il pieno successo del Proof of Concept nel colmare l'anello di congiunzione tra computer vision monoculare ed esecuzione dinamica in simulazione fisica.

```
(base) C:\Users\Alessandro>conda env list

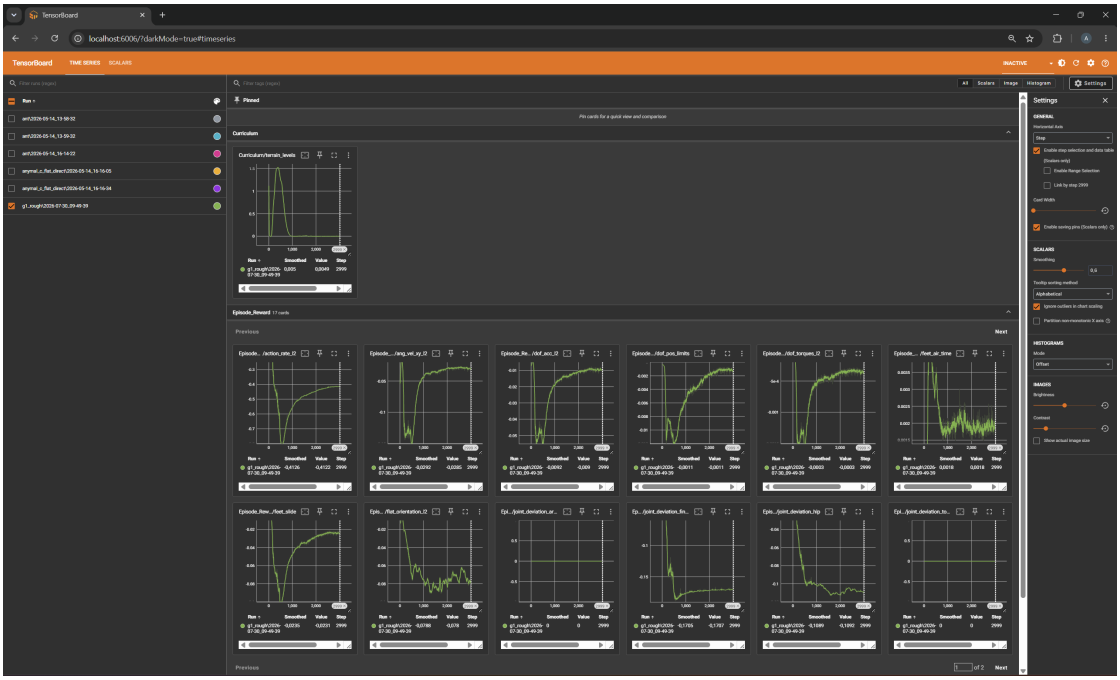
# conda environments:
#
# * -> active
# + -> frozen
base                  *   C:\Users\Alessandro\miniconda3
env_isaaclab              C:\Users\Alessandro\miniconda3\envs\env_isaaclab
freemocap-env             C:\Users\Alessandro\miniconda3\envs\freemocap-env
gemx                      C:\Users\Alessandro\miniconda3\envs\gemx
soma-retargeter           C:\Users\Alessandro\miniconda3\envs\soma-retargeter
base                  *   C:\Users\Alessandro\miniconda3


(base) C:\Users\Alessandro>conda activate env_isaaclab

(env_isaaclab) C:\Users\Alessandro>cd IsaacLab

(env_isaaclab) C:\Users\Alessandro\IsaacLab>tensorboard --logdir logs/rsl_rl
TensorFlow installation not found - running with reduced feature set.
Serving TensorBoard on localhost; to expose to the network, use a proxy or pass --bind_all
TensorBoard 2.20.0 at http://localhost:6006/ (Press CTRL+C to quit)
```

(a) Lancio del server TensorBoard da riga di comando all'interno dell'ambiente `env_isaaclab`.



(b) Dashboard web di TensorBoard con i grafici in tempo reale delle reward di tracking e delle penalità.

Figura 5.9: Monitoraggio dell'addestramento in Isaac Lab tramite TensorBoard: (a) attivazione del server da terminale CLI e (b) interfaccia grafica per l'ispezione delle metriche di apprendimento.

5.7 Analisi Critica: I Limiti dell'Imitazione Umana Pura (Il Collo di Bottiglia Antropomorfo)

A conclusione della validazione del Proof of Concept, emerge una riflessione teorica di primaria importanza per gli sviluppi futuri della robotica umanoide industriale: il ruolo e i limiti del paradigma di imitazione cinematica pura (Pure Motion Mimicry).

5.7.1 Il bias del dato umano e la compressione dei range articolari

L'approccio basato sul costringere un agente robotico a replicare pedissequamente la cinematica umana introduce un intrinseco bias antropomorfo. Il corpo umano è il risultato di compromessi evolutivi e vincoli biologici (struttura ossea, inserzioni muscolari, mobilità legamentosa asimmetrica) che non coincidono necessariamente con l'ottimo meccatronico.

Obbligare un robot a muoversi esattamente come l'operatore ripreso nel video significa imporgli artificialmente i limiti di velocità, flessibilità e stabilità propri dell'essere umano, comprimendo lo spazio delle soluzioni dinamiche che il sistema di controllo potrebbe altrimenti esplorare per massimizzare la robustezza, l'efficienza energetica o la rapidità d'azione.

5.7.2 Le capacità meccaniche inespresse del G1: mobilità del torso e range articolari

L'Unitree G1 presenta caratteristiche fisiche e cinematiche che superano nettamente l'anatomia biologica:

- **Ampia escursione torsionale del torso**: il giunto di imbardata della vita presenta un vincolo meccanico di $\pm 155°$, coprendo un arco utile complessivo di $310°$; una mobilità straordinaria se confrontata con la colonna vertebrale umana, la cui rotazione assiale è fisiologicamente limitata a circa $\pm 40°$;
- **Simmetria e range articolari estesi**: le articolazioni degli arti inferiori e superiori presentano escursioni angolari ampie e densità di coppia elevate erogate da motori brushless dedicati, capaci di generare contromovimenti di recupero dell'equilibrio preclusi alla biomeccanica biologica.

In un paradigma di imitazione pura vincolato ai dati estratti dal video monoculare, queste doti meccaniche avanzate rimangono completamente silenti e inutilizzate, poiché la funzione di reward penalizza qualsiasi configurazione articolare che si discosti dalla posa dell'operatore ripreso.

5.7.3 Prospettiva metodologica: dal motion mimicry puro al controllo ibrido

La sintesi di queste evidenze sperimentali suggerisce una chiara traiettoria evolutiva per l'architettura di controllo: superare la semplice imitazione passiva in favore di un paradigma ibrido basato su prior informati e controllo gerarchico.

In questa visione avanzata:

1. La traiettoria estratta dalla pipeline monoculare (GEM-X + SOMA) non agisce più come un vincolo rigido punto a punto, ma come una guida semantica iniziale (*kinematic prior*);

2. L'ottimizzazione tramite Reinforcement Learning viene guidata primariamente da reward legate al successo del task e alla stabilità (es. manipolazione di un oggetto o utensile, mantenimento dell'orientamento dell'end-effector, minimizzazione delle forze di contatto parassite);

3. La policy di controllo è libera di sfruttare appieno le ridondanze meccatroniche del robot, inclusa la rotazione estesa del torso fino a $\pm 155°$ o estensioni estreme degli arti, per recuperare l'equilibrio a fronte di forti disturbi o per eseguire compiti industriali con velocità ed ergonomia superiori a quelle umane;

4. L'architettura di controllo viene disaccoppiata gerarchicamente in una serie di policy specializzate (*skill policies*), supervisionate da un orchestratore di alto livello (*meta-controller*) incaricato di commutare dinamicamente i comandi ed eliminare i conflitti di ottimizzazione (*reward clashing*).

Un esempio emblematico di questo paradigma, approfondito anche nelle disamine sullo stato dell'arte del controllo per robot umanoidi [5], riguarda la gestione delle perturbazioni critiche: qualora il robot perda l'equilibrio mentre si trova in una postura scomposta o vincolata al task, l'orchestratore interviene tempestivamente sottraendo il controllo al modulo di tracciamento e richiamando una policy dedicata al recupero drastico dell'equilibrio (*fall-recovery policy*). Quest'ultima, svincolata da qualsiasi obbligo di mimesi umana, può sfruttare appieno le doti meccatroniche non convenzionali del robot (come torsioni repentine del busto o passi d'arresto ampi) per neutralizzare la caduta, riaffidando l'esecuzione al task principale non appena ristabilito il bacino di stabilità.

Questo passaggio concettuale proietta la ricerca condotta in questo lavoro oltre il semplice tracciamento video, gettando le basi per un'architettura modulare di intelligenza fisica autonoma realmente ottimizzata per la cinematica del robot umanoide.

Capitolo 6

Valutazione di Fattibilità Industriale, Sviluppi Futuri e Conclusioni

Questo capitolo conclude il lavoro di tesi tirando le somme sui risultati ottenuti e valutando la reale fattibilità dell'introduzione di robot umanoidi in azienda. Vengono presentati i tempi di calcolo dell'intera pipeline, i passi necessari per portare il controllo dal simulatore al robot fisico in sicurezza e le prospettive future più promettenti della robotica moderna. Nelle considerazioni finali si discute infine l'importanza di aver mantenuto l'architettura modulare e di aver affiancato al progetto una Knowledge Base dedicata (LLM-Wiki) per rendere chiaro e immediato il passaggio di consegne all'interno del reparto R&D.

6.1 Sintesi dei Risultati e Bilancio della Ricerca

Il presente lavoro di tesi ha affrontato una delle sfide aperte più rilevanti nel campo della robotica umanoide applicata all'industria: la trasformazione rapida e non invasiva di dimostrazioni umane registrate in video bidimensionali in comandi eseguibili da un robot umanoide.

6.1.1 Traguardi Raggiunti nella Pipeline Video-to-Humanoid

L'indagine scientifica e sperimentale condotta ha consentito di raggiungere i seguenti risultati fondamentali:

1. **Superamento del mocap tradizionale**: Identificazione delle cause pratiche e matematiche che rendono i file `.bvh` da telecamere non calibrate incompatibili con il controllo di un robot umanoide;

2. **Estrazione del movimento da singolo video 2D**: Recupero accurato della posa 3D da una semplice registrazione con webcam o smartphone tramite il modello GEM-X, senza bisogno di calibrazioni ottiche preventive;

3. **Adattamento cinematico sul robot**: Mappatura del corpo umano sulle 29 articolazioni dell'Unitree G1 con SOMA Retargeter, garantendo l'eliminazione dello slittamento dei piedi e il rispetto dei limiti fisici del robot;

4. **Validazione in simulazione dinamica**: Integrazione delle traiettorie in NVIDIA Isaac Lab, verificando che il robot impari a riprodurre il movimento mantenendo stabilmente l'equilibrio dinamico anche su terreni irregolari;

5. **Architettura software modulare**: Organizzazione del progetto in ambienti Conda separati e formati di file standard (`.mp4` → `.csv` → `.usd`), eliminando conflitti tra librerie e rendendo la pipeline replicabile nel tempo.

6.1.2 Dal Mocap Classico alla Visione Neurale Parametrica

Il confronto metodologico tra la fase preliminare (Capitolo 2) e la pipeline definitiva neurale (Capitolo 4) ha evidenziato come l'impiego di modelli corporei parametrici vincolati (SOMA a 77 giunti) sia un prerequisito indispensabile per la robotica.

Mentre la triangolazione geometrica lineare (DLT) soffre di variazioni continue delle lunghezze ossee, occlusioni e rumore cartesiano non differenziabile, i moderni Vision Transformer (ViTPose) combinati con Temporal Transformer a finestra temporale (pari a 120 fotogrammi in GEM-X, equivalenti a 4 secondi di contesto a 30 FPS) consentono di stimare direttamente rotazioni continue nello spazio $SO(3)$. Questo cambio di paradigma ha trasformato un problema mal posto in una catena computazionale deterministica, differenziabile e interamente accelerata su GPU.

6.1.3 Prestazioni e Tempi di Calcolo della Pipeline

Per valutare la sostenibilità computazionale dell'approccio all'interno di un reparto R&D aziendale, i moduli della pipeline sono stati esaminati sulla workstation di sviluppo impiegata nel progetto. L'allocazione della memoria video dedicata (VRAM) e il carico computazionale sono stati monitorati durante le singole fasi di elaborazione tramite il pannello delle prestazioni di sistema (come mostrato a scopo illustrativo in Figura 6.1), registrando i valori di picco raggiunti da ciascun modulo.

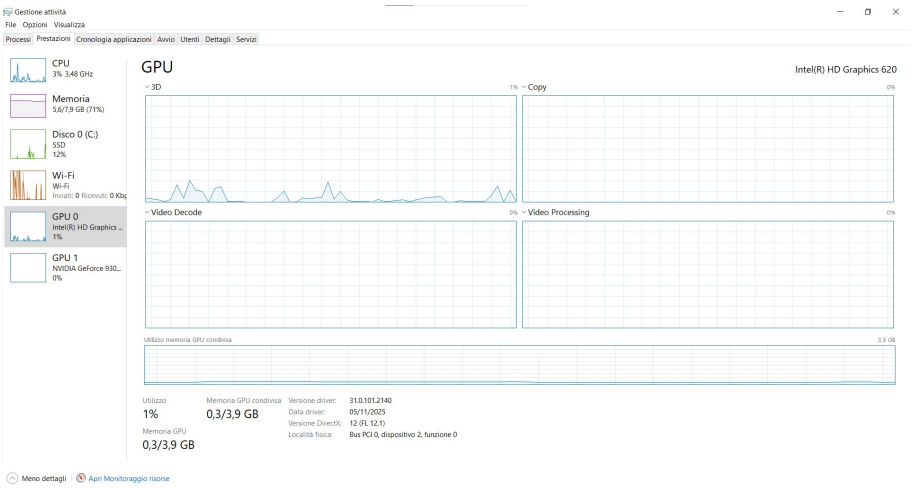


Figura 6.1: Monitoraggio in tempo reale dell'allocazione VRAM e del carico GPU (immagine a scopo illustrativo).

Dall'analisi dei carichi computazionali emerge un principio architetturale di primaria importanza per l'infrastruttura R&D:

- **Collo di bottiglia dell'hardware**: Lo stadio di addestramento dinamico parallelo in NVIDIA Isaac Lab (in modalità headless) rappresenta il processo di gran lunga più intensivo dell'intera pipeline, arrivando a occupare un picco di circa 9.6 GB di VRAM per sostenere la simulazione simultanea di 4096 istanze del robot G1, la dinamica di contatto PhysX 5 e i buffer tensoriali dell'algoritmo PPO (richiedendo una scheda con almeno 16 GB consigliati per operare con adeguato margine);
- **Dimensionamento dell'infrastruttura aziendale**: I processi a monte di visione artificiale (YOLOX, ViTPose, GEM-X) e retargeting cinematico (SOMA) assorbono picchi di memoria più contenuti (rispettivamente circa 3.2 GB, 4.8 GB e 2.1 GB) e operano a velocità comprese tra 15 e 80 FPS. Di conseguenza, il dimensionamento dell'hardware di calcolo aziendale (GPU con 16 GB di memoria video) va tarato prioritariamente sui requisiti della simulazione dinamica in Isaac Lab.

Con questa configurazione, la simulazione massiva raggiunge oltre 67.000 passi fisici al secondo, permettendo di completare l'addestramento dinamico di una policy di locomozione/imitazione su milioni di campioni in circa un'ora e venti minuti di calcolo locale.

In Tabella 6.1 viene riportato il resoconto riassuntivo delle velocità di calcolo, dei picchi di VRAM e delle modalità di esecuzione per ciascuno stadio operativo.

Stadio Pipeline	Modulo / Algoritmo	Velocità di Calcolo	VRAM (Picco)	Esecuzione
Acquisizione Video	Webcam Logitech C920 (1080p)	30 FPS	N/A	Host OS
Rilevamento e Posa 2D	YOLOX + ViTPose	$\approx$ 18.5 FPS	$\approx$ 3.2 GB	GPU
Stima Posa 3D	GEM-X Temporal Transformer	$\approx$ 15.2 FPS	$\approx$ 4.8 GB	GPU
Retargeting su G1	SOMA Retargeter (NVIDIA Warp)	$\approx$ 80 FPS	$\approx$ 2.1 GB	GPU
Addestramento Dinamico	Isaac Lab / PhysX 5 (4096 env)	$\approx$ 67.300 passi/s	$\approx$ 9.6 GB	GPU (Headless)
Inferenza Policy / Play	–	–	–	Robot

Tabella 6.1: Resoconto prestazionale ed esecuzione dei singoli stadi della pipeline.

6.2 Valutazione di Fattibilità Industriale e Impatto per Nexion Group

L'introduzione di robot umanoidi all'interno dei processi produttivi e dei centri di assistenza per autoveicoli rappresenta una transizione strategica per Nexion Group (Corghi). La presente ricerca dimostra che la combinazione di visione artificiale monoculare e simulazione fisica su GPU è in grado di abbattere radicalmente le barriere di accesso alla programmazione robotica.

6.2.1 Flessibilità Operativa e Digitalizzazione Rapida di Tutorial Video

Nella pratica industriale convenzionale, l'insegnamento di una traiettoria articolare complessa a un robot antropomorfo o a un cobot richiede figure specializzate (programmatori PLC e integratori robotici), ore di calibrazione con pulsantiere di autoapprendimento (teach pendant) o l'impiego di tute di teleoperazione immersive non scalabili in officina.

La pipeline sviluppata introduce un cambio di paradigma:

- **Cattura non invasiva**: L'operatore specializzato esegue la lavorazione standard (ad esempio il montaggio o la diagnostica di uno pneumatico su macchinari Corghi) venendo semplicemente ripreso da una telecamera fissa o da uno smartphone;

- **Digitalizzazione dei movimenti**: Il filmato viene convertito automaticamente in traiettorie articolari strutturate (`.csv`), creando un archivio aziendale riutilizzabile di lavorazioni e abilità motorie;
- **Adattamento dinamico automatico**: Attraverso Isaac Lab, la policy di controllo adatta la cinematica ideale alle asperità del suolo e alle perturbazioni fisiche senza richiedere riscritture manuali del codice di basso livello.

6.2.2 La Roadmap Verso il Robot Fisico: Strategie per Colmare il Sim-to-Real Gap

Sebbene il presente lavoro si sia focalizzato sulla validazione end-to-end all'interno dell'ambiente di simulazione, l'architettura è stata esplicitamente concepita per abilitare il trasferimento zero-shot su hardware reale (Unitree G1 fisico). Per colmare definitivamente il divario di realtà (Sim-to-Real gap), la roadmap tecnica per i futuri sviluppi in Nexion prevede tre passaggi chiave:

1. **Modellazione dei limiti fisici dei motori**: Integrazione diretta nelle policy di addestramento delle curve caratteristiche reali dei motori del robot (coppie massime, attriti e saturazioni), assicurando che le traiettorie richieste siano compatibili con la reale potenza degli attuatori;
2. **Domain randomization per la robustezza**: Aggiunta di disturbi casuali durante la simulazione (piccoli ritardi di comunicazione, rumore sui sensori di assetto e variazioni di attrito del suolo), rendendo la policy capace di adattarsi alle inevitabili imperfezioni dell'ambiente reale;
3. **Esportazione ed esecuzione a bordo**: Esportazione della rete neurale addestrata in formati leggeri e ottimizzati (come ONNX), permettendo al computer di bordo del robot di calcolare i comandi in tempo reale e pilotare direttamente i motori fisici.

6.2.3 Sicurezza e Coesistenza Uomo-Robot (pHRI)

L'introduzione di robot umanoidi (con masse di circa 35 kg e motori ad alta densità di potenza) in spazi condivisi con gli operatori d'officina impone un'attenta analisi di sicurezza (physical Human-Robot Interaction, pHRI), per la quale il quadro normativo internazionale (aggiornato con la norma ISO 10218-2:2025 [11], che recepisce e unifica le linee guida collaborative della precedente ISO/TS 15066) è ancora in gran parte aperto e privo di standard dedicati alla locomozione bipedale libera.

Facendo un parallelo con il mondo industriale dei bracci collaborativi (cobot), l'esperienza sul campo dimostra che, nonostante la presenza di sensori di contatto e strategie

per attutire gli urti, i vincoli normativi impongono limiti molto severi sulle forze di impatto. Nella pratica, questo obbliga i cobot a muoversi a velocità estremamente ridotte, costituendo spesso un forte collo di bottiglia per i tempi ciclo e la produttività aziendale.

Nel caso di un robot umanoide bipedale in movimento libero, questo paradigma statico non è direttamente traslabile. Una drastica decelerazione o un arresto impulsivo predefinito rischiano infatti di destabilizzare l'equilibrio della floating base, inducendo ribaltamenti incontrollati che aggraverebbero paradossalmente il potenziale pericolo per l'operatore circostante.

In quest'ottica, la transizione verso un reale impiego industriale richiede di superare il controllo cinematico rigido in posizione a favore di architetture di controllo intrinsecamente cedevoli (*compliant control*) e basate su impedenza o forza-coppia. Sebbene nella validazione in Isaac Lab la regolarizzazione delle accelerazioni e delle derivate di coppia rappresenti un primo passo per smussare le azioni ed evitare comportamenti a scatto, la piena coesistenza in officina richiederà l'integrazione di policy sensibili al contatto. In tale prospettiva, il moto appreso da video monoculare non andrà inteso come un profilo geometrico inderogabile, ma come un riferimento flessibile capace di cedere docilmente o deviare la traiettoria non appena rileva forze di contatto anomale, conciliando l'efficacia operativa con la salvaguardia dell'integrità fisica dell'operatore.

6.3 Prospettive di Ricerca e Tecnologie Emergenti (Physical AI)

Il settore della robotica umanoide e della Physical AI sta attraversando una fase di accelerazione tecnologica esponenziale. Nel corso del 2025 e 2026 sono emersi nuovi paradigmi algoritmici che delineano l'evoluzione naturale della pipeline proposta.

6.3.1 Dalla Cattura Video alla Sintesi Generativa con NVIDIA Kimodo

Nel corso della stesura di questo lavoro di tesi, NVIDIA ha presentato Kimodo [26], un innovativo modello generativo di diffusione per la sintesi e il controllo cinematico del movimento. L'integrazione concettuale di Kimodo negli sviluppi futuri della presente tesi consente non solo di tracciare la rotta evolutiva della ricerca, ma ne valorizza fortemente l'architettura rispetto allo stato dell'arte.

Infatti, mentre Kimodo si focalizza sulla generazione sintetica ex-novo guidata da linguaggio naturale (Text-to-Motion) o vincoli spaziali (Waypoint-to-Motion), esso non affronta la percezione visiva da registrazioni fisiche. La pipeline sviluppata in questa tesi risponde esattamente a tale esigenza fondamentale, consentendo di partire da comuni video

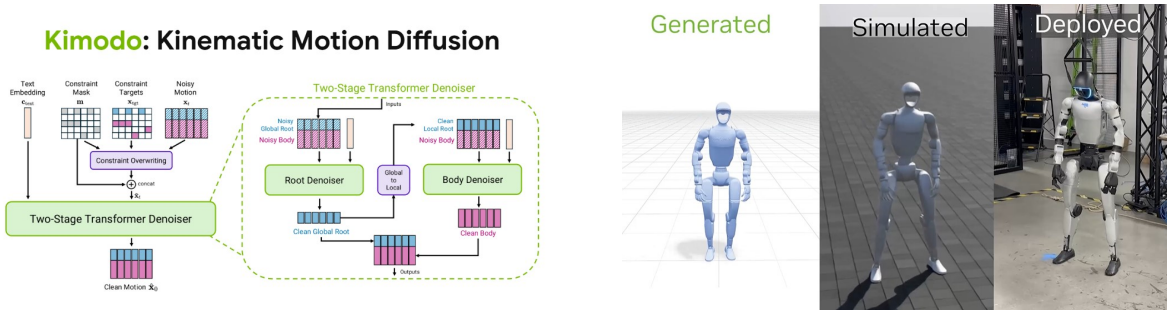
monoculari 2D (Video-to-Motion) di operatori in officina per digitalizzarne fedelmente l'esperienza motoria. In questo modo i due strumenti risultano perfettamente complementari: la nostra catena capitalizza e preserva il patrimonio gestuale reale, mentre Kimodo offre la possibilità di variarlo o espanderlo sinteticamente.

Come illustrato in Figura 6.2(a), Kimodo organizza la generazione del movimento in due fasi sequenziali guidate dalle istruzioni dell'utente e dai vincoli geometrici impostati:

1. **Traiettoria globale** (Root Denoiser): calcola lo spostamento 3D del corpo nello spazio, garantendo che i piedi poggino saldamente a terra senza scivolamenti indesiderati;

2. **Movimento articolare** (Body Denoiser): sintetizza le rotazioni di tutti i giunti del corpo coerentemente con la traiettoria globale e con l'azione descritta nel prompt.

Questo approccio consente di coprire l'intero ciclo di sviluppo (Figura 6.2(b)): dalla generazione cinematica alla simulazione dinamica in Isaac Lab, fino al deployment diretto sul robot fisico reale.



(a) Architettura Transformer a due stadi.

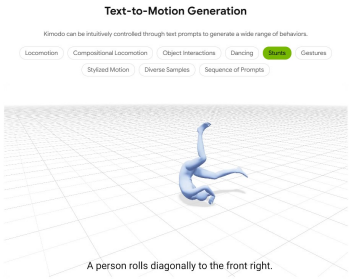
(b) Generazione, simulazione e deploy fisico.

Figura 6.2: Architettura neurale di NVIDIA Kimodo (a) e ciclo completo dal moto sintetizzato al deployment su robot umanoide reale (b) (fonte: [26]).

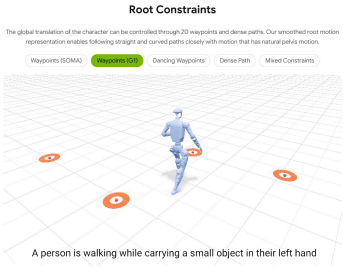
Appartenendo allo stesso ecosistema di ricerca NVIDIA, Kimodo offre una notevole flessibilità con un doppio punto di innesto nella pipeline proposta:

1. **Integrazione a monte (livello umano SOMA)**: Kimodo sintetizza i movimenti direttamente nello scheletro parametrico SOMA a 77 giunti (Figura 6.3(a)). In questa configurazione l'output si inserisce prima del modulo di retargeting, sostituendo il video reale e consentendo al convertitore cinematico di mappare il movimento su qualsiasi robot umanoide tramite file `.bvh` o `.npy`;

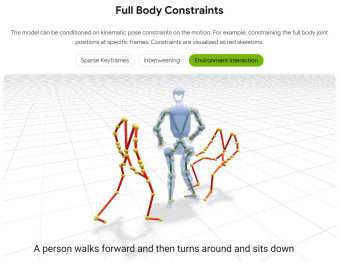
2. **Integrazione a valle (diretta su robot G1)**: Kimodo supporta nativamente anche la generazione di moti già retargettati sul modello cinematico dell'Unitree G1 (Figura 6.3(b-d)). In questo caso il modello scavalca l'ottimizzatore cinematico intermedio, producendo direttamente i dati articolari (`.csv` / `.usd`) pronti per essere forniti a Isaac Lab per l'addestramento dinamico.



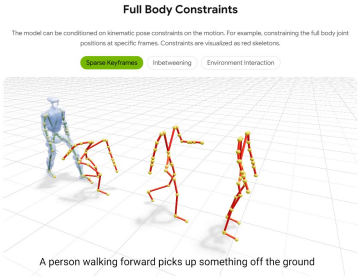
(a) Text-to-Motion puro (SOMA)



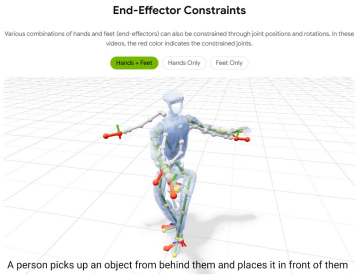
(b) Root Waypoints (G1)



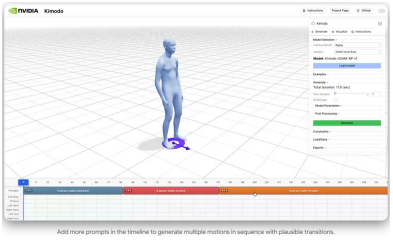
(c) Interazione ambiente (G1)



(d) Full-Body Keyframing



(e) End-Effector Constraints



(f) Timeline interattiva GUI

Figura 6.3: Modalità di condizionamento di NVIDIA Kimodo: dalla generazione da testo su scheletro umano SOMA (a) al controllo vincolato di traslazione radice (b), interazione con ostacoli e arredi (c), pose chiave complete (d), vincoli su organi terminali (e) e composizione sequenziale multi-prompt (f) (fonte: [26]).

Come evidenziato nel carosello di Figura 6.3, l'operatore può guidare la sintesi combinando linguaggio naturale (es. "l'operatore cammina verso lo smontagomme, si piega a 45 gradi e solleva la ruota") con vincoli geometrici su waypoint al suolo (Root Constraints), organi terminali (End-Effector Constraints) o pose chiave (Full-Body Constraints), gestibili anche tramite una comoda timeline grafica (Figura 6.3(f)). Questo paradigma consente all'azienda di generare enormi dataset sintetici per le lavorazioni d'officina senza la necessità di registrare filmati per ogni singola variazione di posa.

6.3.2 Scalabilità del Whole-Body Control e Sim-to-Sim con NVIDIA ProtoMotions

Per addestrare policy in grado di inseguire contemporaneamente migliaia di movimenti eterogenei, il framework NVIDIA ProtoMotions3 [30] rappresenta il riferimento all'avanguardia.

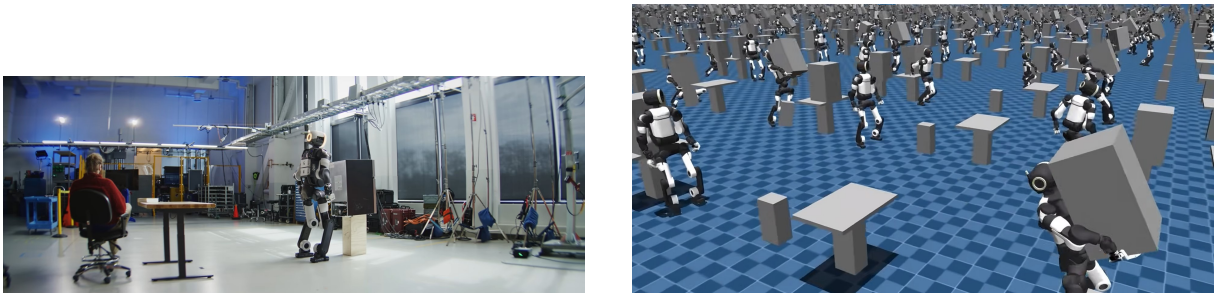
ProtoMotions introduce una metodologia di validazione incrociata multi-simulatore (Sim-to-Sim Evaluation): prima di trasferire una policy sul robot fisico, la rete neurale viene testata su molteplici motori fisici distinti (Isaac Lab con PhysX 5, Newton Physics Engine e MuJoCo [31]). Questo benchmark preventivo permette di identificare ed eliminare eventuali anomalie o sovra-adattamenti alle peculiarità numeriche di un singolo solutore, garantendo una maggiore robustezza nel passaggio alla realtà.

6.3.3 Whole-Body Loco-Manipulation, Retroazione delle Forze e Agilità Sovra-Umana

Negli scenari industriali e di officina, le mansioni operative non si limitano a movimenti statici, ma richiedono la coordinazione simultanea di locomozione e manipolazione bimanuale per il trasporto di carichi pesanti e ingombranti (Whole-Body Loco-Manipulation).

In questo contesto, il controllo puramente cinematico (basato sull'inseguimento delle sole pose visive) risulta insufficiente per via delle elevate forze di contatto e delle variazioni repentine del baricentro. La ricerca attuale si sviluppa lungo due direttrici fondamentali:

- **Sensori di forza indossabili**: l'impiego di guanti tattili o solette sensorizzate durante la cattura dell'operatore umano, permettendo di registrare i profili di forza e pressione esercitati sugli oggetti;
- **Stima di coppia tramite corrente degli attuatori**: sui robot fisici, la retroazione di forza viene ottenuta stimando le coppie esterne a partire dall'assorbimento di corrente dei motori brushless. Questa tecnica richiede una modellazione accurata delle curve caratteristiche dei motori, degli attriti non lineari e del rendimento dei riduttori, parametri che devono essere calibrati e replicati fedelmente in ambiente di simulazione per garantire il successo del trasferimento Sim-to-Real.



(a) Esecuzione su robot Atlas fisico. (b) Addestramento RL parallelo in simulatore.

Figura 6.4: Whole-Body Loco-Manipulation con Boston Dynamics Atlas: confronto tra esecuzione reale nel trasporto di un carico pesante (a) e addestramento per rinforzo in simulazione (b) (fonte: [3]).

Un esempio di riferimento dello stato dell'arte è rappresentato dalla dimostrazione di Boston Dynamics con il robot umanoide elettrico Atlas [3] (Figura 6.4):

1. In simulazione dinamica (Figura 6.4(b)), migliaia di istanze del robot apprendono contemporaneamente tramite Reinforcement Learning a prelevare, trasportare e posizionare un elettrodomestico pesante da un punto A a un punto B, gestendo perturbazioni fisiche, variazioni di massa e attrito;

2. La policy neurale risultante, trasferita direttamente sul robot fisico (Figura 6.4(a)), coordina in tempo reale l'appoggio dei piedi e l'inclinazione del busto in funzione delle forze scambiate con il carico.

Questo paradigma si sposa con il concetto di controller ibridi (motion prior e task-based RL) e con lo sfruttamento integrale dei gradi di libertà non umani del robot. Un robot come l'Unitree G1, potendo ruotare il busto fino a $\pm 155°$ ($310°$ di arco complessivo), è in grado di voltarsi e movimentare carichi pesanti senza dover compiere passi intermedi di riallineamento, superando i limiti fisiologici del corpo biologico in termini di efficienza energetica e stabilità dinamica.

6.3.4 Raccolta Dati su Larga Scala: Setup Egocentrici e Routine Quotidiane

Un trend di grande rilievo nella ricerca internazionale e nell'industria della robotica umanoide (guidato da realtà come Figure AI [4] e dal consorzio Meta Ego-Exo4D [9]) è l'adozione di setup di acquisizione video egocentrici (head-mounted cameras e smart glasses).

Invece di allestire complesse stanze di ripresa con telecamere fisse esterne (exo-view) o tute di cattura dedicate, gli operatori indossano occhiali intelligenti o telecamere frontali leggere per registrare flussi video continui in prima persona (ego-view) durante le normali routine quotidiane o mansioni operative (Figura 6.5).

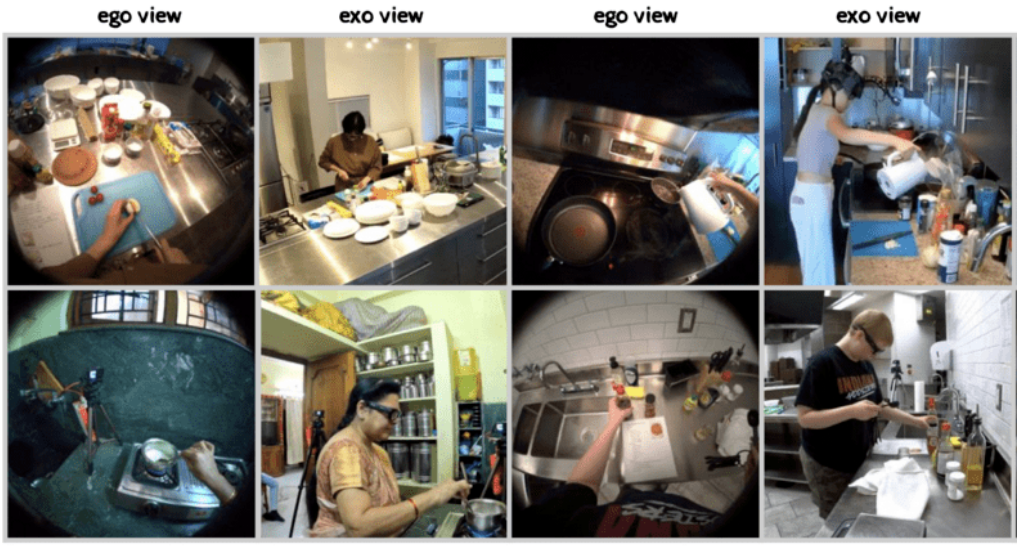


Figura 6.5: Confronto tra prospettiva in prima persona (ego-view) e in terza persona (exo-view) nel benchmark Ego-Exo4D (fonte: [9]).

Questo approccio permette di raccogliere moli enormi di dati visivi e interattivi con costi ridotti e una prospettiva naturale sulle mani dell'operatore. Nel contesto dell'evoluzione

della robotica, questi dataset su larga scala costituiscono una risorsa preziosa per addestrare i futuri modelli di controllo multimodali e generalisti.

6.3.5 Manipolazione Fine con Mani a 5 Dita e Sensori Visuotattili

L'estensione della pipeline alla manipolazione fine (dexterous manipulation) rappresenta un naturale passo successivo. Il framework parametrico SOMA incorpora già 77 gradi di libertà, includendo la cinematica completa delle 5 dita della mano.

Attraverso l'impiego di mani antropomorfe a 5 dita sensorizzate montate sugli avambracci del G1 (come le mani Unitree DexHand o mani con pelle artificiale visuotattile) e la modellazione del contatto continuo con solutori PhysX 5 in Isaac Lab, il robot potrà apprendere compiti delicati come l'inserimento di valvole di pressione, l'avvitamento di dadi ruota e la manipolazione di cavi elettrici.

6.3.6 La Frontiera dei Modelli Vision-Language-Action (VLA / GR00T)

A lungo termine, la convergenza tra modelli linguistici multimodali (VLM) e controllo motorio sta conducendo alla nascita dei modelli Vision-Language-Action (VLA), tra cui l'iniziativa NVIDIA Project GR00T [22].

In questo paradigma unificato, una singola rete neurale di tipo foundation riceve in ingresso flussi visivi dalle telecamere di testa, comandi vocali in linguaggio naturale e letture propriocettive, generando direttamente le coppie articolari per l'intero corpo del robot. La capacità della nostra architettura di produrre dimostrazioni strutturate costituisce il canale di alimentazione primario per il pre-addestramento e il fine-tuning di tali modelli generalisti nei contesti industriali specifici di Nexion Group.

6.4 Considerazioni Conclusive e Trasferimento del Know-How

Il lavoro di tesi svolto si è confrontato con uno spettro tecnologico e scientifico eccezionalmente vasto ed eterogeneo, spaziando attraverso discipline di frontiera spesso frammentate: dalla computer vision 2D avanzata (YOLOX, ViTPose) alla cinematica epipolare multicamera, dal 3D lifting temporale neurale (GEM-X) ai modelli corporei unificati (SOMA), dall'ottimizzazione quadratica su GPU compilata al volo (NVIDIA Warp) fino all'apprendimento per rinforzo massivo parallelo in simulazione dinamica tensorizzata (NVIDIA Isaac Sim / Isaac Lab).

6.4.1 La Modularità Architetturale come Valore Ingegneristico

Per padroneggiare tale complessità senza incorrere in rigidi vincoli proprietari o fragilità di integrazione (vendor lock-in), l'intera infrastruttura software è stata concepita con un'architettura rigorosamente modulare e disaccoppiata.

Grazie alla separazione ermetica dei quattro ambienti Conda isolati e all'adozione di standard aperti universali (`.mp4`, `.bvh`, `.csv`, `.usd`, `.onnx`), ciascun singolo componente della catena può essere aggiornato, testato o integralmente sostituito in futuro (ad esempio rimpiazzando GEM-X con un nuovo modello di visione o passando da Isaac Lab a un nuovo simulatore) a patto di rispettare i formati di interfaccia tabulari e cinematici consolidati. Questa indipendenza funzionale garantisce all'azienda la massima libertà di sperimentazione tecnologica nel lungo periodo.

In questo contesto, un'ulteriore considerazione ingegneristica per il trasferimento industriale riguarda il regime di licenza dei software adottati. Sebbene la pipeline impieghi ampiamente framework open-source e librerie aperte rilasciate con licenze permissive (quali Apache 2.0, MIT o BSD per PyTorch, FreeMoCap, SOMA Retargeter e Isaac Lab), anche l'open-source è soggetto a precisi vincoli legali. Nel passaggio da una fase di Proof of Concept a un eventuale rilascio produttivo su scala aziendale, il reparto R&D è chiamato a verificare costantemente i vincoli di conformità (licensing compliance), distinguendo tra codice liberamente integrabile e modelli o dataset soggetti a clausole restrittive per sola ricerca (research-only) o specifici accordi EULA (come per NVIDIA Omniverse / Isaac Sim).

6.4.2 Superamento dell'Attrito Aziendale tramite la LLM-Wiki

Tuttavia, all'interno di un reparto aziendale di Ricerca e Sviluppo, una complessità tecnologica così elevata rischia spesso di generare forti barriere di trasmissione: per chi legge la tesi o per colleghi ingegneri chiamati a replicare gli esperimenti, verificare i parametri di addestramento o estendere il codice su nuovi robot, l'eterogeneità delle librerie e dei framework matematici può indurre dubbi, colli di bottiglia e dispersione delle competenze.

Per risolvere strutturalmente questo problema e garantire un trasferimento fluido, persistente e affidabile del know-how, l'intero percorso di ricerca è stato affiancato e documentato tramite una LLM-Wiki strutturata secondo l'architettura a tre livelli di Andrej Karpathy (dettagliata nell'Appendice A):

1. **Base di conoscenza persistente e versionata**: Oltre 150 tra concetti scientifici, schede sorgente dei paper e manuali tecnici sono stati formalizzati all'interno di un Knowledge Graph locale in formato Markdown/Obsidian;

2. **Oracolo interattivo per l'R&D**: Grazie all'interconnessione densa dei link bidirezionali (`[[wikilinks]]`) e ai protocolli di tracciabilità (`log.md` e script di linting automatico), qualsiasi membro del team può interrogare direttamente l'infrastruttura per comprendere le scelte architetturali, reperire frammenti di codice o rigenerare documentazione tecnica aggiornata;

3. **Asset permanente e incrementale**: La Wiki non costituisce una fotografia statica della tesi, ma un sistema vivente che continuerà ad accogliere nuovi paper, modelli emergenti e log di test, preservando l'investimento intellettuale e accelerando i progetti futuri dell'azienda.

6.4.3 Nexion Group e la Frontiera della Physical AI

In conclusione, questo lavoro di tesi attesta la concreta fattibilità dell'impiego di robot umanoidi nei contesti operativi e industriali moderni, dimostrando come la visione artificiale e la simulazione dinamica possano trasformare semplici registrazioni video in controllo motorio avanzato.

Attraverso la visione pionieristica della direzione R&D di Nexion Group e la guida strategica dell'Ing. Lorenzo Vezzani e dell'Ing. Bruno Ferrarini, il polo di Correggio si posiziona all'avanguardia nell'esplorazione della Physical AI, tracciando una rotta solida e metodologicamente rigorosa verso la fabbrica intelligente e la robotica collaborativa del futuro.

Appendice A

Architettura della LLM-Wiki e Knowledge Graph

In questa appendice viene sintetizzata l'architettura della LLM-Wiki ispirata al paradigma di Andrej Karpathy [12], concepita per strutturare la conoscenza tecnica preservando il contraddittorio scientifico tra fonti discordanti, e l'estensione applicativa sviluppata per l'integrazione di una pipeline locale in LaTeX dedicata alla produzione di documentazione e reportistica aziendale.

Tale metodologia non costituisce unicamente un proof-of-concept a sé stante, ma riveste un'importanza tecnica strategica per il progetto di tesi. Essa ha consentito di creare una vera e propria Wiki locale altamente specializzata, consultabile con estrema precisione e livello di dettaglio grazie all'assimilazione ("ingestion") strutturata della vasta bibliografia di settore. Inoltre, l'infrastruttura così concepita si propone come uno strumento duttile e riutilizzabile in scenari futuri: fornendo in ingresso nuovi documenti, il sistema si converte in un motore autonomo per la stesura e la produzione documentale avanzata.

A.1 Architettura a Livelli, Comandi Operativi e Contraddittorio Nativo

L'architettura della LLM-Wiki si basa sulla netta separazione strutturale tra dati grezzi e conoscenza compilata (Figura A.1):

- **Livello `raw/` (immutabile)**: deposito delle fonti primarie originali fornite dall'utente. Questo strato è a sola lettura e non viene mai modificato dal modello. È suddiviso in sottocartelle tipiche:
  - **`raw/papers/`**: articoli scientifici e pubblicazioni tecniche (PDF, Markdown);
  - **`raw/notes/`**: appunti sperimentali, note di laboratorio e verbali tecnici;

  - **`raw/web/`**: articoli, specifiche e pagine web acquisite da browser;
  - **`raw/assets/`**: figure, schemi originali e documentazione grafica.

- **Livello `wiki/` (strutturato e persistente)**: rappresenta la base di conoscenza interamente curata e compilata in autonomia dall'LLM. L'agente mantiene la coerenza e l'interconnessione di questo strato distillando le evidenze grezze estratte da `raw/`, organizzandole in schede atomiche:
  - **`wiki/sources/`**: schede di sintesi analitica di ciascuna fonte letta;
  - **`wiki/concepts/`**: pagine dedicate a teorie, formulazioni matematiche e metodologie;
  - **`wiki/entities/`**: schede relative a librerie software, piattaforme fisiche e ambienti di calcolo;
  - **`wiki/syntheses/`**: rassegne dello stato dell'arte e matrici comparative cross-documentali.

- **Mappa topologica del vault (`index.md`)**: è il catalogo centrale del sistema, costantemente sincronizzato. Invece di dover scansionare decine di file per comprendere la struttura del vault (operazione che saturerebbe rapidamente la finestra di contesto), l'agente consulta preliminarmente `index.md`, che elenca ogni pagina creata con la tipologia e un riassunto sintetico in una sola riga. Funziona quindi come entry-point rapido a bassissimo costo di token, indicando dove reperire un'informazione o dove innestare nuovi collegamenti.

- **Registro transazionale (`log.md`)**: file in modalità append-only in cui l'agente annota in ordine cronologico l'esito di ogni operazione svolta. Funziona come una vera e propria "scatola nera" di audit per l'utente, consentendo di verificare a colpo d'occhio quali pagine siano state create o modificate dall'agente e monitorare la crescita della conoscenza, senza dover essere ricaricato nella finestra di contesto durante la normale consultazione.

A.1.1 Comandi Operativi Fondamentali

L'interazione e la manutenzione della base di conoscenza avvengono mediante quattro comandi principali:

- **`ingest <nome_file>`**: istruisce l'agente a processare una fonte grezza presente in `raw/` (eseguendo l'estrazione testuale o l'OCR di documenti PDF e immagini), creare la relativa scheda analitica in `wiki/sources/` e aggiornare a cascata da 5 a 15 pagine di concetti correlati, innestando collegamenti bidirezionali ([[wikilinks]]);

- **`query`**: interroga la base di conoscenza su quesiti specifici o confronti metodologici; l'agente risponde contestualizzando le evidenze e citando le pagine wiki correlate;
- **`synthesize`**: richiede la compilazione di matrici comparative o capitoli di sintesi globale a partire dai concetti presenti nel vault, salvando il risultato in `wiki/syntheses/`;
- **`lint`**: esegue un controllo di integrità ed health-check dell'intero vault, individuando collegamenti interrotti (broken links), pagine orfane o concetti mancanti.

Un punto di forza nativo di questo approccio è la conservazione del contraddittorio scientifico: mentre i modelli linguistici convenzionali tendono ad appiattire le divergenze producendo risposte mediatrici generiche, la struttura atomica a grafo della Wiki mantiene distinte le conclusioni discordanti tra fonti diverse, mappando i limiti di validità di ciascuna evidenza sperimentale.

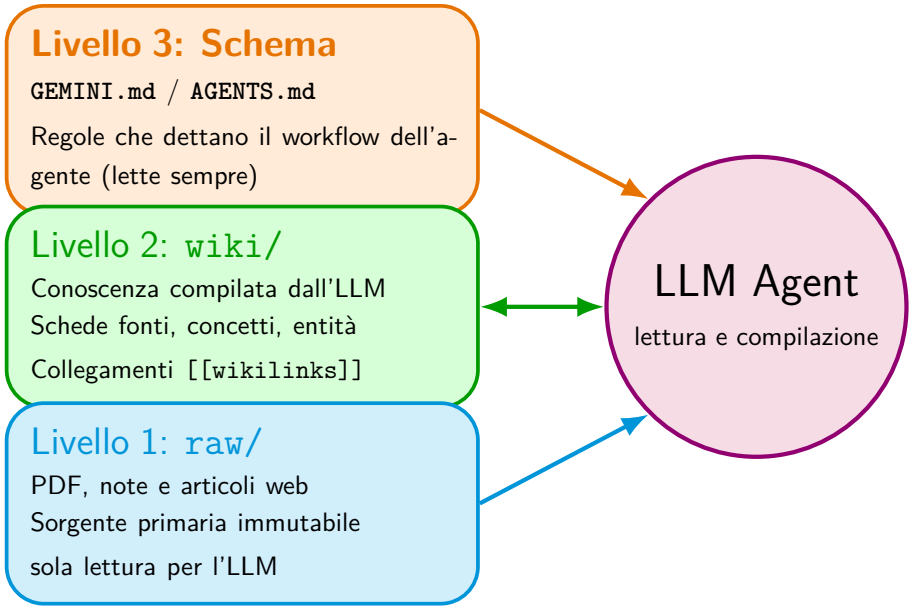


Figura A.1: Architettura a tre livelli della LLM-Wiki (paradigma di Karpathy).

A.2 Governo Cognitivo: `GEMINI.md`, `SKILL.md` e Memoria con `log.md`

La governance operativa dell'agente si struttura su componenti complementari di controllo:

- **Regole di sistema e specifica di schema** (`GEMINI.md` e `SCHEMA.md`): definiscono la governance operativa. In particolare, `GEMINI.md` fissa il protocollo comportamentale permanente dell'agente (letto a ogni interazione: immutabilità di `raw/`, effetto a cascata, compounding), mentre `SCHEMA.md` formalizza l'esatta struttura dei file: i template frontmatter YAML per ciascuna categoria e le regole sintattiche

dei collegamenti bidirezionali ([[wikilinks]]), garantendo piena compatibilità con il Knowledge Graph e con il plugin Dataview di Obsidian.

- **Abilità modulari** (`SKILL.md`): racchiudono procedure specialistiche ed esecutive (es. compilazione LaTeX, audit di coerenza). Vengono caricate on-demand (Progressive Disclosure) esclusivamente quando il task lo richiede, evitando di appesantire inutilmente la finestra di contesto.

- **Memoria episodica** (`log.md`): poiché gli LLM sono nativamente privi di memoria tra sessioni distinte, `log.md` funge da registro cronologico persistente. Tracciando ogni operazione eseguita (`ingest`, `query`, `lint`), consente all'agente di consultare lo storico delle modifiche e mantenere continuità decisionale senza ricorrere a complessi vector database esterni.

A.3 Estensione Applicativa: Modelli Open-Weight, Integrazione LaTeX e Modularità delle `SKILL.md`

La LLM-Wiki si dimostra uno strumento di straordinaria efficacia per le attività di Ricerca e Sviluppo (R&D), poiché consente di consolidare e interrogare grandi volumi di letteratura scientifica e documentazione tecnica complessa senza dispersione di contesto. Un vantaggio metodologico cruciale di questo paradigma risiede nella sua totale indipendenza tecnologica: il sistema non richiede di fidelizzarsi a una specifica intelligenza artificiale commerciale proprietaria, ma può essere interamente costruito e alimentato anche mediante modelli open-weight (eseguibili su infrastrutture locali). L'unico requisito essenziale è che tali modelli siano sufficientemente performanti nelle capacità di ragionamento e dotati di adeguate finestre di contesto (context window) per processare documenti densi e schemi articolati.

Partendo da tale presupposto, si è scelto di estendere la filosofia originale di Karpathy, tradizionalmente orientata alla sola consultazione ipertestuale, esplorando la possibilità di integrare nativamente LaTeX in locale, al fine di accelerare e automatizzare la produzione di documentazione tecnica, whitepaper e report aziendali ad elevato rigore formale, matematico e bibliografico.

Per realizzare tale estensione in modo modulare ed efficiente, sono state definite delle procedure specialistiche formalizzate come `SKILL.md`, collocate nella cartella locale `.agents/skills/`:

- **Caricamento on-demand**: a differenza delle regole di base lette in modo permanente a ogni interazione, le abilità modulari vengono invocate ed elaborate dall'agente esclusivamente nel momento in cui si richiede l'effettiva stesura di un documento;

- **Specifiche e standard aziendali**: ciascuna scheda di abilità racchiude le linee guida formali, i vincoli tipografici, la suddivisione modulare delle sezioni e le convenzioni per formule, figure e tabelle conformi agli standard aziendali definiti;
- **Compilazione deterministica**: l'agente estrae la conoscenza strutturata dal vault Markdown, genera i relativi file `.tex` e si interfaccia direttamente con i compilatori locali (pdflatex, bibtex) per produrre deliverable in formato PDF pronti all'uso.

A.4 Visualizzazione Topologica tramite Obsidian Knowledge Graph

La struttura a collegamenti bidirezionali ([[wikilinks]]) consente di ottenere una rappresentazione grafica immediata e navigabile di tutti i concetti compressi e strutturati dall'LLM attraverso l'IDE Obsidian (Figure A.2 e A.3).

In particolare, come illustrato nella Figura A.3a, posizionando il cursore sopra un nodo di interesse l'interfaccia evidenzia istantaneamente tutte le connessioni a cui è legato, permettendo di consultare in modo trasparente la documentazione scientifica e risalire direttamente ai paper/file correlati.

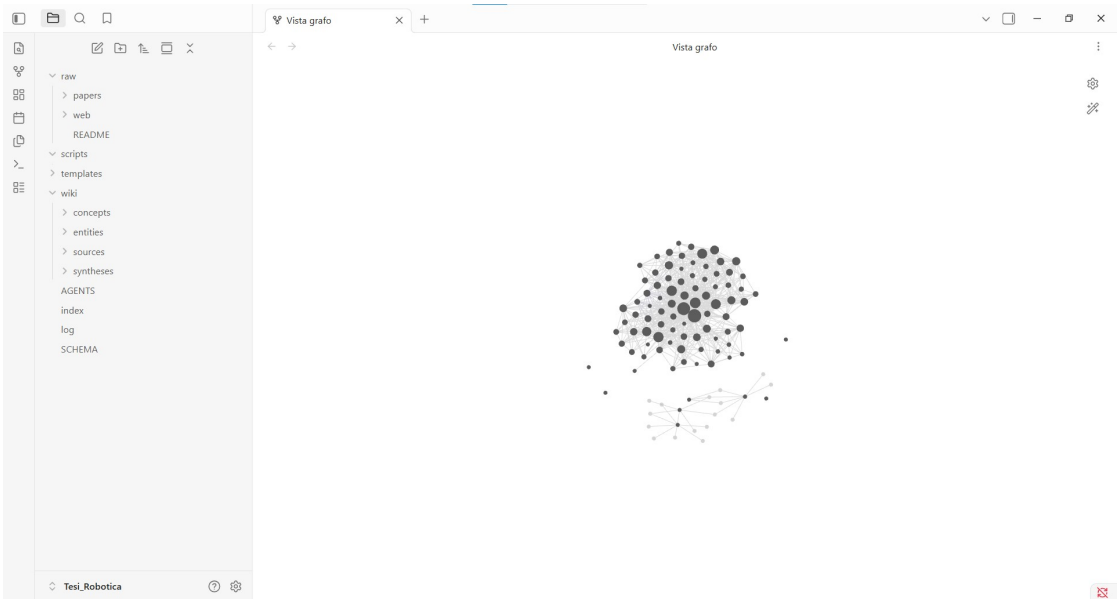
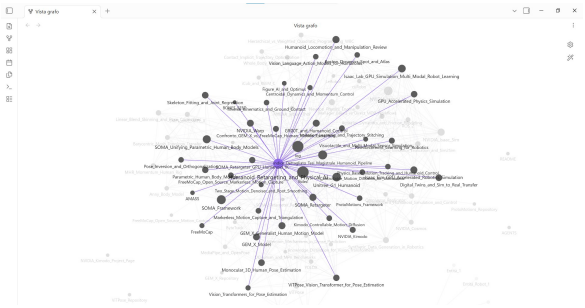
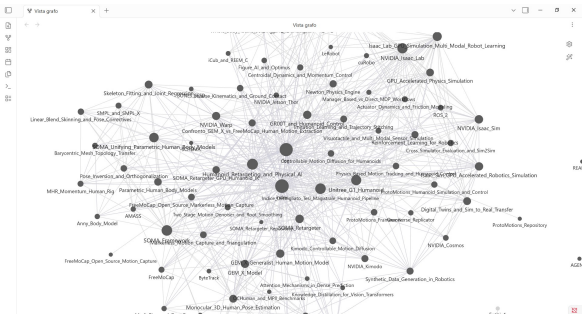


Figura A.2: Panoramica globale del Knowledge Graph in Obsidian.



(a) Evidenziazione dei collegamenti su un nodo selezionato.



(b) Dettaglio locale tra schede sorgente e concetti.

Figura A.3: Dettagli del Knowledge Graph: (a) connessioni attive del nodo selezionato; (b) interconnessioni atomiche locali.

Bibliografia

[1] Anaconda Inc. Anaconda Software Distribution and Conda Package Management System. https://docs.anaconda.com/, 2024. [Online; documentazione software]. [Consultato il: 08-01-2026].

[2] Nikodem Bartnik. I Tried To Build a Robot Like Boston Dynamics With Isaac Sim (Training AI to Walk in Simulation). https://www.youtube.com/watch?v=YDzHL2JSCHc, 2025. [Video online]. Dimostrazione sperimentale di reward hacking su robot quadrupede tipo Ant (min. 12:55). [Consultato il: 15-01-2026].

[3] Boston Dynamics. How Does Atlas Learn? | Inside the Lab: Whole-Body Loco-Manipulation via Reinforcement Learning. https://www.youtube.com/watch?v=xKK5ze3FukQ, 2026. [Video online]. Dimostrazione sperimentale del trasporto carichi in officina tramite RL. [Consultato il: 10-02-2026].

[4] Figure AI, Inc. Figure: Autonomous General-Purpose Humanoid Robots. https://www.figure.ai/, 2024. [Online]. Specifiche della piattaforma umanoide autonoma. [Consultato il: 18-01-2026].

[5] Foundation Robotics. How We Train Humanoid Robots: Multi-Policy Orchestration and Low-Level Motor Skills. https://www.youtube.com/watch?v=oRymQyooWjE, 2026. [Video online]. Trattazione sulle architetture di controllo gerarchiche e policy motorie. [Consultato il: 20-02-2026].

[6] Sicong Gao, Maurice Pagnucco, Tomasz Bednarz, and Yang Song. NVIDIA Isaac Sim: Enabling Scalable, GPU-Accelerated Simulation for Robotics. *arXiv preprint arXiv:2606.03551*, 2026. Disponibile su: https://arxiv.org/abs/2606.03551.

[7] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. YOLOX: Exceeding YOLO Series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. Disponibile su: https://arxiv.org/abs/2107.08430.

[8] Google Developers. MediaPipe Pose Landmark Detection Guide and Topology. https://developers.google.com/mediapipe/solutions/vision/pose_landmarker, 2024. [Online; documentazione ufficiale]. Google AI. [Consultato il: 12-01-2026].

[9] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, et al. Ego-Exo4D: Understanding Skilled Human Activity from First- and Third-Person Perspectives. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19383–19400, 2024. Disponibile su: https://ego-exo4d-data.org/.

[10] Zhaoyuan Gu, Junheng Li, Wenlan Shen, Wenhao Yu, Zhaoming Xie, Stephen McCrory, Xianyi Cheng, Abdulaziz Shamsah, Robert Griffin, C. Karen Liu, et al. Humanoid Locomotion and Manipulation: Current Progress and Challenges in Control, Planning, and Learning. *arXiv preprint arXiv:2501.02116*, 2025. Disponibile su: https://arxiv.org/abs/2501.02116.

[11] International Organization for Standardization. ISO 10218-2:2025 - Robotics — Safety Requirements for Robot Systems in an Industrial Environment — Part 2: Robot Systems, Robot Applications and Robot Cells Integration. Standard internazionale, International Organization for Standardization, Ginevra, Svizzera, 2025. [Norma tecnica]. Disponibile su: https://www.iso.org/standard/83547.html.

[12] Andrej Karpathy. LLM Knowledge Bases and Structured Wiki Workflows. https://github.com/karpathy, 2026. [Online]. Framework concettuale su basi di conoscenza strutturate esterne e flussi wiki per LLM. [Consultato il: 05-01-2026].

[13] Louis Le Lay. Kinova Gen3 Sim-to-Real Deployment with NVIDIA Isaac Lab. https://github.com/louislelay/kinova_isaaclab_sim2real, 2024. [Online; repository open-source]. Trasferimento Sim-to-Real di policy RL su manipolatore Kinova. [Consultato il: 25-01-2026].

[14] Jiefeng Li, Jinkun Cao, Haotian Zhang, Davis Rempe, Jan Kautz, Umar Iqbal, and Ye Yuan. GENMO: A GENeralist Model for Human MOtion (GEM-X). *arXiv preprint arXiv:2505.01425*, 2025. Disponibile su: https://arxiv.org/abs/2505.01425. Codice: https://github.com/NVlabs/GEM-X.

[15] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *ACM Transactions on Graphics (TOG)*, 34(6):248:1–248:16, 2015. Disponibile su: https://doi.org/10.1145/2816795.2818013.

[16] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. MediaPipe: A Framework for Building Perception Pipelines. *arXiv preprint arXiv:1906.08172*, 2019. Disponibile su: https://arxiv.org/abs/1906.08172.

[17] Jon Matthis and The FreeMoCap Contributors. FreeMoCap: An Open-Source, Markerless Motion Capture System. https://freemocap.org/, 2026. [Online; software open-source]. Piattaforma per il tracciamento del movimento markerless. [Consultato il: 10-01-2026].

[18] Mayank Mittal, Yunrong Guo, Pascal Roth, David Hoeller, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac Lab: A GPU-Accelerated Simulation Framework for Multi-Modal Robot Learning. *arXiv preprint arXiv:2511.04831*, 2025. Disponibile su: https://arxiv.org/abs/2511.04831.

[19] NVIDIA Corporation. Closing the Sim-to-Real Gap: Training Spot Quadruped Locomotion with NVIDIA Isaac Lab. https://developer.nvidia.com/blog/closing-the-sim-to-real-gap-training-spot-quadruped-locomotion-with-nvidia-isaac-lab/, 2024. [Online; blog tecnico]. Integrazione tra Isaac Sim per la dinamica/rendering e Isaac Lab per il Robot Learning. [Consultato il: 22-01-2026].

[20] NVIDIA Corporation. Isaac Lab Documentation: Task Workflows (Manager-Based and Direct Workflows). https://isaac-sim.github.io/IsaacLab/main/source/overview/core-concepts/task_workflows.html, 2024. [Online; documentazione tecnica]. Specifiche architetturali dei paradigmi per il Robot Learning in Isaac Lab. [Consultato il: 20-01-2026].

[21] NVIDIA Corporation. NVIDIA Warp: A High-Performance Python Framework for Differentiable Spatial Computing and GPU Physics Simulation. https://github.com/NVIDIA/warp, 2024. [Online; repository software]. [Consultato il: 10-01-2026].

[22] NVIDIA Corporation. Project GR00T: A Generalist Foundation Model for Humanoid Robot Learning. https://www.nvidia.com/en-us/research/robotics/gr00t/, 2024. [Online]. NVIDIA Physical AI Foundation Model Initiative. [Consultato il: 15-01-2026].

[23] NVIDIA Robotics Research. SOMA Retargeter: GPU-Accelerated Human-to-Humanoid Motion Retargeting with Ground Contact and Joint Limits. https://github.com/NVIDIA/soma-retargeter, 2026. [Online; software toolkit]. Framework di cinematica inversa differenziabile basato su NVIDIA Warp. [Consultato il: 15-02-2026].

[24] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala.

PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 8024–8035, 2019. Disponibile su: https://doi.org/10.48550/arXiv.1912.01703.

[25] Pixar Animation Studios and Alliance for OpenUSD. Universal Scene Description (OpenUSD): Specification and Core Architecture. https://openusd.org/, 2024. [Online; specifiche di standard aperto]. [Consultato il: 14-01-2026].

[26] Davis Rempe, Mathis Petrovich, Ye Yuan, Haotian Zhang, Xue Bin Peng, Yifeng Jiang, Tingwu Wang, Umar Iqbal, David Minor, Michael de Ruyter, Jiefeng Li, Chen Tessler, Edy Lim, Eugene Jeong, Sam Wu, Ehsan Hassani, Michael Huang, Jin-Bey Yu, Chaeyeon Chung, Lina Song, Olivier Dionne, Jan Kautz, Simon Yuen, and Sanja Fidler. Kimodo: Scaling Controllable Human Motion Generation. Technical report, NVIDIA Research, March 2026. [Online]. Disponibile su: https://research.nvidia.com/labs/sil/projects/kimodo/. [Consultato il: 12-03-2026].

[27] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. In *Conference on Robot Learning (CoRL)*, pages 91–100. PMLR, 2022. Disponibile su: https://proceedings.mlr.press/v164/rudin22a.html.

[28] Jun Saito, Jiefeng Li, Michael de Ruyter, Miguel Guerrero, Edy Lim, Ehsan Hassani, Roger Blanco Ribera, Hyejin Moon, Magdalena Dadela, Marco Di Lucca, Qiao Wang, Xueting Li, Jan Kautz, Simon Yuen, and Umar Iqbal. SOMA: Unifying Parametric Human Body Models. *arXiv preprint arXiv:2603.16858*, March 2026. Disponibile su: https://arxiv.org/abs/2603.16858.

[29] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*, 2017. Disponibile su: https://arxiv.org/abs/1707.06347.

[30] Chen Tessler, Yifeng Jiang, Xue Bin Peng, Erwin Coumans, Yi Shi, Haotian Zhang, Davis Rempe, Gal Chechik, and Sanja Fidler. ProtoMotions3: An Open-Source Framework for Humanoid Simulation and Control. https://github.com/NVlabs/ProtoMotions, 2025. [Online; codice sorgente]. NVIDIA Research. [Consultato il: 15-02-2026].

[31] Emanuel Todorov, Tom Erez, and Yuval Tassa. MuJoCo: A Physics Engine for Model-Based Control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5026–5033. IEEE, 2012. Disponibile su: https://doi.org/10.1109/IROS.2012.6386109.

[32] Unitree Robotics. Unitree G1 Humanoid Robot: Technical Specifications and URDF/USD Kinematics Model. https://www.unitree.com/g1, 2024. [Online; specifiche tecniche]. [Consultato il: 15-01-2026].

[33] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. ViTPose: Simple Vision Transformer Baselines for Human Pose Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 38571–38584, 2022. Disponibile su: https://doi.org/10.48550/arXiv.2204.12484.

[34] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. In *European Conference on Computer Vision (ECCV)*, pages 1–21, 2022. Disponibile su: https://doi.org/10.1007/978-3-031-20047-2_1.

Ringraziamenti

Desidero esprimere la mia più sincera gratitudine a tutte le persone che hanno reso possibile la realizzazione di questo lavoro di tesi e che mi hanno accompagnato lungo il mio percorso universitario e personale.

Desidero ringraziare il mio Relatore, il Prof. Riccardo Rubini, per avermi riaccompagnato anche in questo traguardo dopo aver già guidato la mia tesi triennale. Lo ringrazio per la rinnovata fiducia nell'affidarmi la responsabilità di un progetto così stimolante, per avermi fatto sentire sempre appoggiato per qualsiasi dubbio e per aver reso possibile l'opportunità di svolgere questo percorso all'interno di Nexion Group.

Ringrazio calorosamente i miei Correlatori aziendali, il Dir. Ing. Lorenzo Vezzani e l'Ing. Bruno Ferrarini, insieme a tutto il reparto di Ricerca e Sviluppo di Nexion Group, per avermi accolto con entusiasmo, per aver creduto in questo progetto e per avermi fornito risorse, competenze e piena fiducia per condurre la sperimentazione su piattaforme robotiche avanzate. Questa esperienza ha rappresentato per me una straordinaria opportunità di crescita professionale e umana.

Un grazie immenso va alla mia famiglia: a mia madre, a mio padre e a mio fratello Simone, per essere stati il mio punto fermo e un sostegno incondizionato in ogni momento. Un pensiero e un ringraziamento speciale vanno ai miei nonni, e in particolare al nonno Giuseppe, per noi nipoti da sempre nonno Peppe, venuto a mancare in questi mesi durante il mio percorso di studi: so che sarai sempre con me. Questo traguardo appartiene tanto a me quanto a tutti voi.

Un ringraziamento speciale va alla mia fidanzata, per essermi stata sempre accanto con amore, pazienza e costante incoraggiamento lungo questo percorso non facile e spesso stressante: so che potrò sempre contare su di te.

Ringrazio di cuore tutti i compagni di corso e colleghi con cui ho condiviso gli anni universitari tra lezioni, esami e lunghi progetti pratici. Un ringraziamento speciale va alla Squadra Genius, i miei amici di sempre: il vostro affetto e le risate condivise hanno reso questo percorso unico e leggero.

Infine, grazie a chiunque abbia condiviso con me anche solo un piccolo tratto di questo cammino, regalandomi un consiglio, un sorriso o un ricordo prezioso.