Humanoid Network: A Decentralized Motion Data Network for Physical AI

Humanoid Network Foundation

contact@humanoidnetwork.org

Legal Disclaimer. Nothing in this White Paper is an offer to sell, or the solicitation of an offer to buy, any tokens, securities, or other financial instruments. This White Paper is provided for informational purposes only in connection with receiving feedback and comments from the public. Any offering of tokens (or other instruments) would be made only pursuant to definitive offering documents, which would include full disclosure of material information and risk factors.
Nothing in this White Paper should be treated or read as a guarantee or promise of how Humanoid Network's business, protocol, network, software, or any token will develop, or of the utility or value of any token. No allocation, vesting schedule, total supply, or emission rate for any token is disclosed or committed to in this paper; all such parameters remain subject to ongoing design and governance.
This White Paper outlines current plans, which may change at its discretion, and the success of which will depend on many factors outside Humanoid Network's control, including market conditions, regulatory developments, and changes in technology and in the robotics and AI industries, among others. Any statements about future events are based solely on Humanoid Network's analysis of the issues described in this White Paper. That analysis may prove to be incorrect.
See Section B for a more complete statement of the legal characteristics of the **$HAN** token.

Contents

Abstract

Physical AI is bottlenecked by motion data. Robot demonstrations are expensive to collect; egocentric human video offers a scalable alternative but faces two structural problems: collecting the data across jurisdictions requires per-country payroll infrastructure, and the motions produced by generative models must be verified as physically executable before they are useful for robot training. Humanoid Network addresses both problems in a single protocol.

We introduce the Humanoid Network, a decentralized motion data network built on three deeply integrated subsystems. **HanVerse** is a global, contributor-sourced dataset of egocentric human motion captured through Aria-class wearables, smartphone rigs, and partner industrial rigs, with over 1,300 hours of annotated manipulation and locomotion data and a continuously growing corpus. **Hydra-Gen** is a motion World Foundation Model: a two-stage transformer diffusion model trained on 700 hours of optical motion capture that generates controllable humanoid motion from natural language and a comprehensive suite of kinematic constraints (full-body keyframes, sparse joint positions and rotations, 2D waypoints, dense paths), reaching 282M parameters trained on 16 GPUs at batch size 2048. **Hydra-Track** is a universal humanoid tracking policy: a reinforcement-learning controller trained on 100M frames over 9k GPU hours that supersizes motion tracking across network size (1.2M–42M parameters), data (100M frames), and compute, and achieves 100% success on 50 diverse real-world trajectories on the Unitree G1 humanoid in zero-shot sim-to-real transfer. The three subsystems compose: prompts and constraints flow into Hydra-Gen, candidate motions flow through Hydra-Track on a simulated humanoid, and the validated outputs are attested by an Oracle layer that produces signed proof records for eligibility, provenance, and enterprise verification.

The **$HAN** token coordinates access to the protocol economy after the canonical HAN token is created on Base at TGE. Humanoid Network's own protocol infrastructure remains separate: veHAN locks can represent non-transferable participation power, Motion Credits can expand testnet-safe submission capacity, proof registries can record Hydra validation outcomes, and future reward controllers can stay disabled until reserve unlocks, app capacity, scoring, anti-abuse controls, and approvals are complete. By replacing per-country employment with proof-gated protocol coordination, the Humanoid Network enables contributions from individuals, academic labs, and industry partners worldwide without jurisdictional friction. By adding physics validation between data collection and eligibility records, it ensures that useful contributions are not just recorded but evaluated for robot learning. Together, these properties produce a transition economy that aligns the workers most affected by physical AI automation with the motion data that trains the next generation of embodied intelligence.

1 Introduction

Large-scale imitation learning has enabled policies to handle broader task distributions, more visual variation, and longer horizons, echoing trends seen in large vision and language models [1–3]. However, unlike those domains, robot learning faces a fundamental bottleneck: collecting robot demonstrations requires physical hardware, expert teleoperation, and controlled setups. As a result, expanding robot datasets in scale and diversity remains slow, expensive, and difficult to sustain.

In contrast, egocentric human data offers a promising alternative. Humans naturally perform manipulation and locomotion tasks across diverse environments on a daily basis, generating rich behavioral data at a scale that is infeasible for robots alone [4, 5]. Importantly, human data also provides a unifying abstraction for the community. Instead of coordinating around a specific robot embodiment, researchers can focus on curating diverse, real-world experience data while deferring embodiment decisions downstream. This property has driven growing academic and commercial interest in leveraging egocentric human demonstrations, supported by recent advances in wearable sensors [6, 7] and large-scale data capture systems [8].

Despite this promise, two major challenges remain. First, effective human-to-robot transfer remains an open research problem, with unresolved questions around the embodiment gap and scaling behavior [9, 10]. Second, most existing human datasets are one-off, static releases collected for a specific study, making further scaling difficult [7, 8]. Addressing these limitations requires more than collecting a larger dataset: it calls for a continuously growing data ecosystem that can evolve with new contributors and provide durable insights into human-to-robot transfer.

1.1 The scaling bottleneck is operational, not scientific

Recent work has validated that co-training robot policies with egocentric human data produces clear and reproducible performance improvements across multiple labs, tasks, and robot embodiments [9–11]. The science of cross-embodiment transfer is maturing rapidly. What has not been solved is the operational infrastructure for collecting this data at global scale.

Current approaches to large-scale egocentric data collection depend on institutional coordination. Academic consortia hire demonstrators through university channels, open bank accounts in each jurisdiction, manage payroll across dozens of labor markets, and negotiate data-sharing agreements between institutions. This model has produced impressive results, but it hits three structural walls:

Jurisdictional payment complexity. Compensating contributors in 50 countries requires 50 sets of employment law compliance, banking relationships, and tax withholding procedures. The overhead scales linearly with geographic reach, creating a ceiling on contributor diversity that is determined by operational capacity rather than scientific need.

Demographic concentration. When participation is gated by institutional affiliation, contributor populations cluster in wealthy countries with strong research universities. This introduces systematic bias into the data: the environments, objects, cultural practices, and physical morphologies represented in the dataset reflect a narrow slice of global diversity.

No sustained incentive alignment. In one-off collection campaigns, contributors have no long-term incentive to improve quality, return for additional tasks, or develop expertise in demonstration techniques. Each campaign starts from scratch, losing the accumulated skill and institutional knowledge of prior participants.

1.2 Web3 as proof and coordination infrastructure

Token-based coordination provides a jurisdictionless rail for proof, access, and future protocol eligibility. A contributor in Lagos, Bangalore, or Sao Paulo can submit quality demonstrations through the same proof pipeline as a contributor in San Francisco. No local bank integration is required for the protocol to record provenance, HYDRA scores, novelty checks, and eligibility state.

This is not Web3 for the sake of Web3. It is a pragmatic choice driven by the specific constraints of global data collection. Traditional systems are weak at portable contribution identity, transparent proof records, and cross-border participation state. Public rails can make those records inspectable while keeping claim execution disabled until the protocol is ready.

1.3 Contributions

This paper makes the following contributions:

1. **HanVerse protocol and platform.** A complete system for collecting, processing, validating, and distributing egocentric human demonstration data at global scale, combining phone-based capture (HANCAPTURE), cloud data management (HANDB), automated quality scoring (HANSCORE), and token-incentivized contributor coordination.

2. **A protocol coordination layer for data collection.** A framework for aligning Contributor, veHAN participant, and Enterprise buyer incentives through quality-weighted eligibility, physics-validated proof gating, lock-weighted participation, and clear separation between the canonical HAN token and Humanoid Network's protocol contracts.
3. **The displaced labor thesis.** An economic argument for why the workers most affected by physical AI automation are the ideal contributors to robot training data, and how proof-gated protocol participation can create a transition economy that benefits both sides of the automation equation.
4. **Cross-embodiment transfer validation.** Building on established findings that human egocentric data improves robot policy performance [9, 11], we describe how HanVerse data feeds into validated human-to-robot transfer pipelines across multiple robot platforms.

2 The Physical AI Labor Transition

The relationship between automation and labor is not new, but physical AI introduces a distinctive pattern that creates a natural demand for HanVerse.

2.1 Task reallocation, not wholesale replacement

The most useful framework for analyzing how technology changes labor demand is the task model developed by Acemoglu and Autor [12]. Their central insight is that labor market changes cannot be understood through broad skill categories alone; they must be understood through the assignment of specific tasks to labor or capital. Acemoglu and Restrepo extend this by distinguishing the displacement effect of automation (machines performing tasks previously done by workers) from the reinstatement effect (new tasks pulling labor back into production) [13]. More recent work sharpens the point: technological change can augment labor, augment capital, automate tasks previously allocated to labor, or create new tasks, each with different consequences for employment, wages, and factor shares [14].

Physical AI represents the next frontier of this reallocation. Where agentic software is displacing coordination-heavy knowledge work (document processing, customer operations, procurement), embodied systems are beginning to automate manipulation, logistics, and service tasks in the physical world. The workers currently performing these tasks, warehouse pickers, line cooks, caregivers, assembly operators, are precisely the population most qualified to demonstrate them for robot learning.

2.2 Contributors as transition workers

HanVerse creates a reinstatement channel in the Acemoglu framework: data demonstration as a new task category that absorbs labor displaced by automation. A warehouse picker who loses shifts to an automated picking system retains deep expertise in the manipulation strategies, failure recovery patterns, and environmental awareness that robot policies need to learn. By demonstrating these tasks through HanCapture, displaced workers convert their embodied expertise into proofed training data that can be evaluated for future protocol eligibility.

This is not charity or make-work. The data has real economic value to enterprises training robot systems, and the contributors' domain expertise produces higher-quality demonstrations than untrained participants. The incentive alignment is genuine: the people who know the tasks best are the ones generating the data, and the protocol can record proof of that contribution regardless of where they live.

2.3 Why this requires decentralized infrastructure

The displaced worker population is globally distributed, often concentrated in countries without robust digital payment infrastructure, and frequently working informally. Traditional employment channels cannot reach them at scale. Opening a subsidiary or payroll account in Indonesia, Nigeria, or the Philippines for every small demonstration task is economically irrational for any single data collection campaign. But a contributor in any of those countries can use a mobile wallet to maintain portable contribution identity, proof records, and future protocol eligibility.

The expected eligibility weight for a contributor i submitting demonstrations across tasks can be expressed as:

$$\mathbb{E}[W_i] = \sum_{t \in \mathcal{T}_i} q(d_{i,t}) \cdot w(t) \cdot \alpha(\text{div}_i), \tag{1}$$

where $q(d_{i,t})$ is the quality score of the demonstration for task t, $w(t)$ is the task weight, and $\alpha(\text{div}_i)$ is a diversity multiplier based on the contributor's demographic and geographic novelty relative to the existing corpus. The diversity multiplier naturally makes the network more valuable to underrepresented populations, since their contributions fill gaps that the dataset needs most.

3 System Architecture

The Humanoid Network is organized into six layers, illustrated in Fig. 1. Data flows from contributors through capture into the platform, where it is processed, scored, and passed through the HYDRA physics layer before the ORACLE records provenance and eligibility attestations. Enterprise customers access processed, physics-verified data through the platform layer; contributors build proofed participation state through the protocol layer.

3.1 Architecture principles

The architecture follows five design principles:

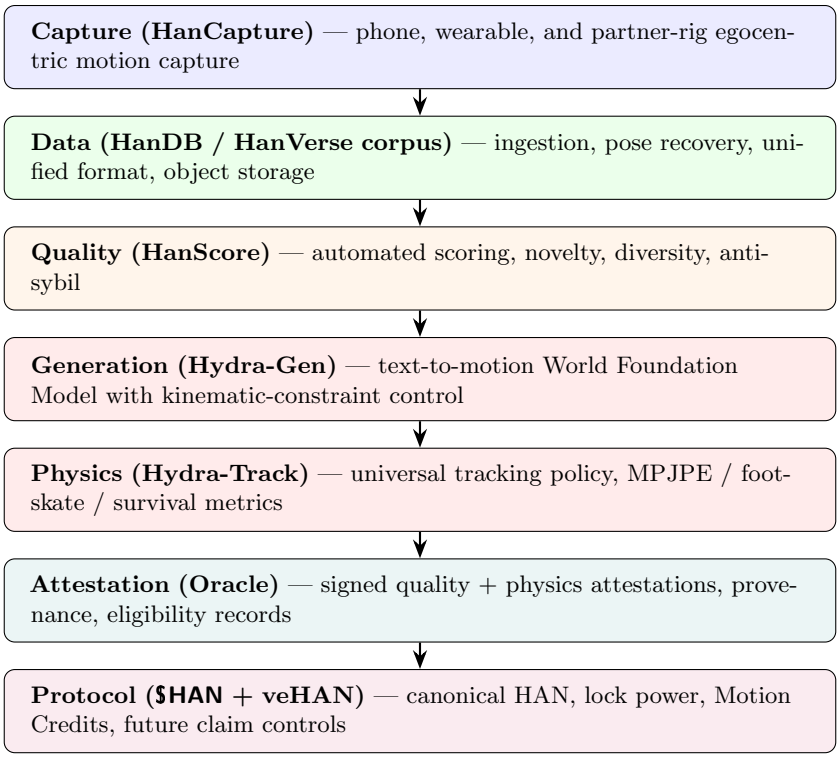
Off-chain compute, on-chain proof. Latency-sensitive operations (motion capture, ML inference, physics simulation, quality scoring) run off-chain. Provenance commitments, physics-validation attestations, eligibility records, and future governance actions can touch the blockchain. This keeps the system responsive while anchoring trust to a public ledger.

Multi-source capture by design. No single collection channel can cover the full distribution of environments, embodiments, and contact dynamics. The system accepts phone-based capture, wearable capture, and partner-sourced data through a unified ingestion pipeline.

Physics validation between data and eligibility. Every reward-eligible motion is executed on a simulated humanoid inside the HYDRA layer before it affects protocol eligibility. Physically unrealizable motions (fall within the first second, foot-skate above a spam threshold, unrecoverable tracking failure) are rejected or flagged. This ensures the dataset is not just large but useful for robot policy learning.

Quality as a first-class primitive. Every episode is scored before it can affect eligibility or dataset trust. Quality scoring is not a post-hoc filter; it is the mechanism that determines corpus composition, future protocol weight, and enterprise confidence.

Jurisdictionless contributor access. Any person with an approved account and supported wallet can participate when the product lane is open. Onboarding requires no institutional affiliation, no bank account, and no employment contract.



Enterprise customers access the physics-verified dataset through the Data layer; proof and eligibility state flow through the Protocol layer.

Figure 1: System architecture. Seven layers: Capture, Data, Quality, Generation, Physics, Attestation, Protocol. Data flows downward. Every reward-eligible contribution passes through quality scoring, generation-and-retrieval, and physics validation before the ORACLE signs a proof record. Reward claims remain disabled until separately approved.

3.2 Actors

The Humanoid Network defines five actor classes. **Contributors** supply egocentric demonstration data and generated robot motions that can pass HYDRA proof gates. **Reviewers and proof systems** run automated and human-in-the-loop quality checks, including novelty, duplicate, and physics-execution review. **veHAN participants** may lock **$HAN** for non-transferable participation power when the lock vault is live, but the v0 lock vault does not include a reward payout function. **Enterprise buyers** purchase dataset bundles, streaming subscriptions, and evaluation runs through the platform API. **Protocol guardians and future governance participants** manage approved parameters, feature gates, and upgrade paths subject to launch, legal, and security controls.

4 HanVerse: The Global Motion Dataset

HANVERSE is the data layer of the Humanoid Network: a continuously growing, contributor-sourced corpus of egocentric human manipulation and locomotion data, structured for direct use in robot policy training. Unlike prior static human-video datasets [4, 5, 15], HANVERSE is built to evolve with new contributors, new embodiments, and new task distributions over time.

4.1 Two-component structure

HanVerse is split into two complementary components:

HanVerse-A ("Academic") covers a small number of tightly specified *flagship* tasks collected under a shared protocol across academic labs. It is designed for cross-lab reproducibility: the same six manipulation tasks, the same quality constraints, the same annotation schema, the same evaluation protocol. This layer enables controlled study of human-to-robot transfer where confounds (task definition drift, scene drift, demonstrator protocol drift) are minimized.

HanVerse-I ("Industry") aggregates data from industry partners running their own capture pipelines in diverse indoor environments and task distributions. It is scale-focused: orders of magnitude larger than the academic layer, covering thousands of unique open-ended tasks. Industry partners ship denser annotations (fine-grained language descriptions at 1–2 second intervals, active-hand flags, static-vs-mobile manipulation tags) that are well suited to language-conditioned policy training such as vision-language-action models [2, 16].

The composition of the dataset at writing is summarized in Table 1. The total corpus contains 1,362 hours of egocentric motion data across 240 scenes, 1,965 tasks, and 2,087 demonstrators. HanVerse-A contributes curated multi-lab flagship-task data; the two largest HanVerse-I partners contribute the bulk of the open-ended corpus. New partners and new demonstrators onboard continuously through the contributor flow described in Section 10.

Table 1: HanVerse composition at writing. The academic layer is small by design (controlled flagship tasks); the industry layer provides open-ended scale.

Component	Share	Hours	Episodes	Tasks
HanVerse-A (all academic partners)	5.5%	75	2,385	6 (flagship)
HanVerse-I partner A	76.1%	1,035	72,993	1,898
HanVerse-I partner B	18.4%	250	3,128	45
Total	100.0%	**1,362**	**78,506**	**1,949**

Within HanVerse-I, the task distribution (Table 2) is heavily weighted toward everyday manipulation categories that intersect directly with commercial robotics roadmaps: logistics and warehousing (15.4%), cooking (13.7%), cleaning (11.6%), laundry (10.9%), hardware and assembly (6.8%), crafts (4.0%), and gardening (3.2%). The remainder spans miscellaneous household and service-sector activities.

Table 2: HanVerse-I task category distribution by hours.

Category	Share	Hours	Category	Share	Hours
Logistics	15.4%	209	Hardware/Assembly	6.8%	92
Cooking	13.7%	186	Crafts	4.0%	54
Cleaning	11.6%	158	Gardening	3.2%	44
Laundry	10.9%	148	Misc./Open-ended	34.4%	466

4.2 Capture pipeline

HanVerse collects data through multiple hardware systems, illustrated in Fig. 2. Regardless of source, every episode lands in a unified format containing egocentric video, hand keypoints, and camera poses.

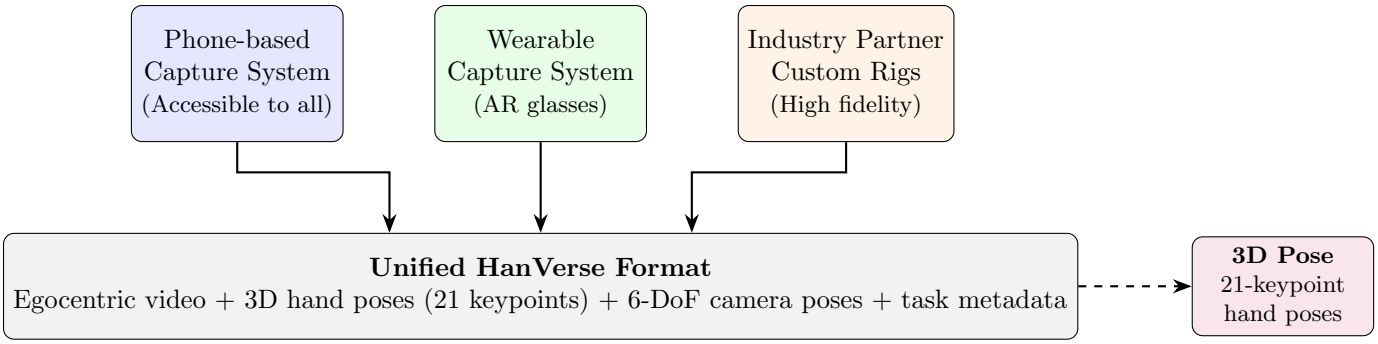


Figure 2: Human data capture setup. HANVERSE collects data through a variety of hardware systems, including a phone-based capture system (accessible to everyone), wearable devices (AR glasses for academic and partner labs), and custom setups by industry partners. Regardless of source, human data is processed into a unified format that contains at minimum egocentric videos, hand keypoints, and camera poses.

4.2.1 Phone-based capture system

To broaden access to data capture, the primary HANVERSE collection pathway uses commodity smartphones. The HANCAPTURE app uses the ultrawide camera recording egocentric RGB video at 1080p and 30 FPS. The phone is positioned to approximate a head-mounted perspective, providing a first-person view of manipulation activities.

Captured videos are uploaded to the HANDB cloud platform, where a processing pipeline recovers 6-DoF head pose via visual tracking and estimates 3D hand poses with 21 keypoints per hand. The app provides real-time guidance on camera angle, lighting conditions, and frame coverage to ensure captured data meets minimum quality thresholds.

Each captured episode $\mathcal{E}_i$ is represented as:

$$\mathcal{E}_i = (V_i,\ H_i,\ P_i,\ \tau_i,\ \kappa_i,\ \sigma_i), \tag{2}$$

where V_i is the video stream, H_i is the estimated 3D hand pose sequence (21 keypoints per hand per frame), P_i is the 6-DoF camera pose trajectory, τ_i is the task identifier, κ_i is the contributor identifier, and σ_i is the cryptographic submission signature for provenance.

4.2.2 Wearable capture system

For academic labs, HANVERSE standardizes on Project Aria-class wearable AR glasses [6] as the primary capture platform. The device integrates a wide-field-of-view RGB camera with two synchronized monochrome scene cameras used for SLAM and hand tracking, plus an inertial measurement unit for visual-inertial odometry. Machine Perception Services are used to produce calibrated 6-DoF head pose, temporal alignment across cameras, and 3D hand pose estimation. The side cameras are critical: they maintain visibility of hand motion even when hands move out of the forward-looking RGB view, producing richer annotation coverage than phone-based capture alone.

4.2.3 Industry partner rigs

HANVERSE-I partners operate custom head-mounted capture rigs in diverse indoor environments. A representative rig uses a pair of fisheye RGB cameras in a coplanar collinear stereo configuration with a fixed 6 cm baseline, recording at 1920×1200 resolution at 30 FPS, supplemented with depth sensing and an IMU. Pose estimation runs a 3D hand-pose pipeline with 21 keypoints per hand.

Partner data enters HanDB through the same ingestion pipeline and is normalized to the unified episode format.

4.2.4 Human data annotation

HanVerse augments egocentric human demonstrations with structured annotations tailored for robot policy learning. For each frame, the pipeline estimates 3D hand pose for both hands using 21 keypoints per hand in the camera frame, paired with a calibrated 6-DoF head pose from visual-inertial SLAM or model-based pose estimation.

Beyond poses, annotation granularity differs across components. HanVerse-A follows a lightweight per-episode protocol: task description, scene identifier, primary manipulated objects, demonstrator metadata. HanVerse-I provides denser annotations including fine-grained (1–2 second) language descriptions, active-hand indicators, static-versus-mobile manipulation flags, and additional contextual tags where available. Figure 3 illustrates the dual-level annotation structure used at dataset scale: a high-level overview description per clip, timeline-segmented fine-grained descriptions per sub-action, plus LLM paraphrases that increase lexical diversity without distorting semantics.

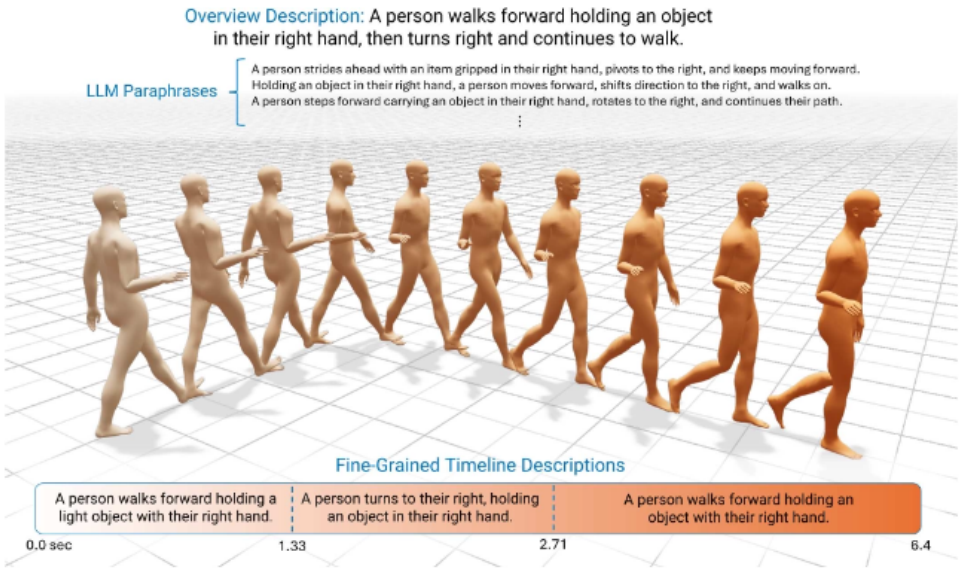


Figure 3: Dual-level annotation in HanVerse-I. Each motion clip carries (top) a high-level overview description, (bottom) timeline-segmented fine-grained descriptions per sub-action, and LLM paraphrases of both that expand lexical diversity while preserving semantics. The annotation schema is what enables fine-grained text conditioning in Hydra-Gen (Section 6).

4.3 Flagship tasks (HanVerse-A)

The academic component ships with six shared flagship tasks that every participating lab collects under a common protocol. The tasks are chosen to span the major manipulation regimes encountered in household and logistics environments, and are summarized in Table 3. Each task is collected in "dataset units" — roughly five minutes of recording per unit, 5–10 demonstrations per unit — with the workspace standardized to approximately 40 cm × 60 cm, hand and object visibility enforced through capture guidance, and demonstrator identity, scene, and object set logged with each unit.

Table 3: HANVERSE-A flagship tasks. Every academic lab collects all six under a shared protocol to enable cross-lab reproducibility.

Task	Manipulation	Specification
object-in-container	Single-arm	40 s continuous pick / place / dump of objects between containers.
cup-on-saucer	Bimanual, precision	Reorient a cup with both hands and place it on a saucer.
bag-grocery	Bimanual, deformable	Open a cloth grocery bag with one hand, load 1–3 items with the other.
fold-clothes	Bimanual, deformable	Three-fold a T-shirt starting from random initial configuration.
scoop-granular	Single-arm, tool use	Scoop dry granular material (e.g. beans) into a container until full.
sort-utensils	Single-arm, precision	Pick and sort mixed utensils by category into labeled bins.

Within HANVERSE-A each lab contributes 8–12 distinct scenes per task and 1–10 dataset units per scene, with 1–8 demonstrators across the network. Up to 30 distinct objects are used per task within a single lab; independent object procurement across labs introduces substantial cross-site variation in object geometry, appearance, and material. The same instruction set is given to every demonstrator; the consistent differences between demonstrators (in motion timing, coordination strategy, and hand trajectory) are retained as a first-class signal rather than suppressed.

4.4 Controlled-diversity subsets

Global aggregation gives realistic diversity but uneven coverage across scenes and demonstrators. To enable controlled study of diversity effects, HANVERSE maintains secondary datasets where data are allocated via structured assignment matrices that decouple scene diversity from demonstrator diversity so each can be independently scaled and studied. For `cup-on-saucer` and `fold-clothes` the academic partners collected such matrices against a fixed pool of 16 demonstrators $\times$ 16 scenes, supporting three experimental regimes used in Section 9:

- **Single-scene demonstrator scaling.** Fix a single scene; scale the number of training demonstrators across $\{1, 2, 4, 8, 16\}$ while holding the total time budget at 2 hours (so per-demonstrator duration shrinks proportionally). Evaluation is on held-out demonstrators.
- **Multi-scene demonstrator scaling.** Fix 8 scenes and scale the demonstrator count across $\{4, 8, 12\}$ at an 8-hour budget. Evaluation is on held-out demonstrators across all 8 scenes.
- **Scene diversity scaling.** Fix a demonstrator pool and scale the scene count across $\{1, 2, 4, 8, 16\}$ under a family of data-density fractions $\{6.25\%, 12.5\%, 25\%, 50\%, 100\%\}$ relative to a 60-min-per-scene baseline. Evaluation is on unseen scenes and unseen demonstrators.

These matrices are the empirical substrate for the diversity findings reproduced in Section 9: scene diversity is the dominant driver of generalization to novel environments under limited budgets, while demonstrator diversity is the dominant driver for generalization across unseen human embodiments.

4.5 Visual coverage versus robot data

An important empirical observation about HanVerse-I is that its visual coverage is substantially broader than both the academic layer and single-robot-lab data of comparable size. UMAP projections of DINOv3 embeddings for `fold-clothes` show robot-only data clustered tightly around a single laboratory's environment statistics, HanVerse-A spread across several cluster centers corresponding to the participating labs, and HanVerse-I filling the manifold between and beyond them. The practical implication is that a policy trained on HanVerse-I is exposed to a much wider slice of lighting, surfaces, clutter, and camera placement than a policy trained on any single lab's robot corpus — which matters for deployment beyond lab-specific appearance statistics.

4.6 HanDB: scalable data management

To support protocol-wide collaboration and long-term dataset growth, HanVerse develops HanDB, a cloud-based system for continuous ingestion, processing, and access of heterogeneous human and robot data (Fig. 4). Data from distributed sources are uploaded to object storage and converted into a unified, training-ready format shared across the HanVerse ecosystem.

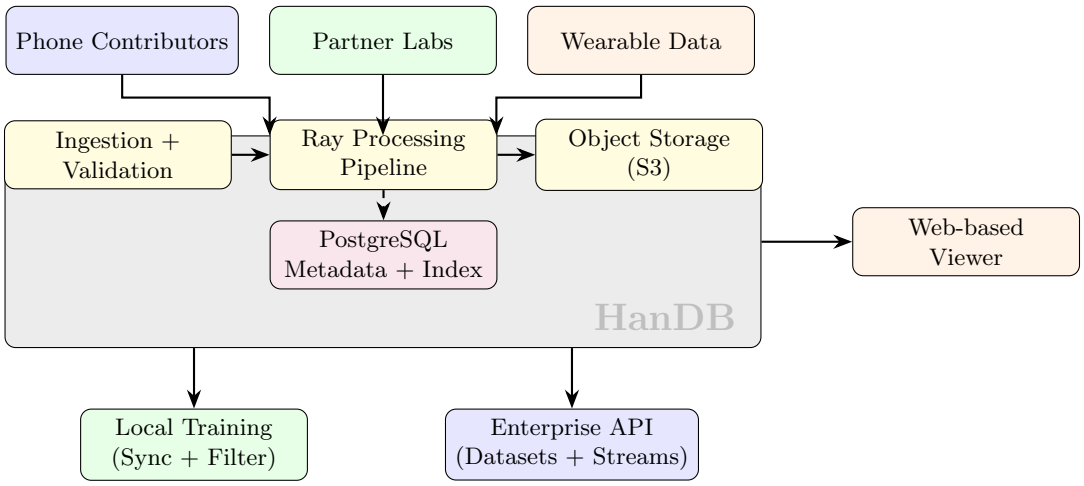


Figure 4: HanDB architecture. Human data from phone contributors, partner labs, and wearable sources are ingested into a cloud-based processing pipeline, unified in a common storage format, and made accessible through a web-based viewer. Users can sync filtered subsets of the dataset to local machines for downstream policy training.

A nightly pipeline performs standardized preprocessing, validation, and indexing to ensure consistent usability for downstream learning. Episode metadata are registered in a centralized SQL database, enabling structured queries over tasks, scenes, data sources, and annotation types. HanDB also provides a web-based interface for browsing demonstrations, inspecting annotations, and tracking dataset growth. For local training, users can synchronize filtered subsets of the dataset via configuration files, enabling reproducible access without manual data management.

The HanDB episode table schema is detailed in Table 4. Fields in italics are populated or updated during automated processing.

The processing pipeline operates in six stages for each submitted episode:

1. **Format validation and ingestion.** Verify video encoding, resolution, frame rate, and metadata completeness. Reject malformed submissions before entering the processing queue.

Table 4: Schema of the HANDB episode table. Fields in italics are updateable during automated processing.

Field	Description
`episode_hash`	Unique identifier for the episode, derived from UTC timestamps at upload time.
`contributor_id`	Identifier for the human contributor or demonstrator.
`source`	Data collection site or partner lab.
`task`	Canonical task name (e.g., *bag-groceries*, *fold-clothes*).
`embodiment`	Embodiment type (human, robot platform, etc.).
`robot_name`	Robot platform identifier, if applicable.
num_frames	Number of frames in the processed trajectory (updated post-processing).
`task_description`	Free-form natural language description of the task.
`scene`	Scene or environment identifier.
`objects`	Serialized list of objects involved in the episode.
processed_path	S3 path to processed data artifacts (updated post-processing).
processing_error	Error message logged during automated processing, if any.
`mp4_path`	S3 path to rendered visualization video, if available.
`is_deleted`	Flag indicating whether the episode has been removed or deprecated.
`quality_score`	Composite quality score $Q(\mathcal{E})$ from HANSCORE pipeline.
`submission_sig`	Cryptographic submission signature σ_i for provenance.

2. **Provenance verification.** Check the cryptographic submission signature σ_i against the contributor's registered device key. Verify timestamp plausibility and device attestation where available.
3. **Pose estimation.** Recover 6-DoF camera pose via visual-inertial SLAM or visual tracking. Estimate 3D hand poses (21 keypoints per hand) using model-based pose estimation.
4. **Feature extraction.** Run object detection, action segmentation, and scene classification. Extract visual embeddings for downstream novelty computation.
5. **Quality scoring.** Compute the composite quality score $Q(\mathcal{E}_i)$ (Section 5), including task relevance, demonstration quality, novelty, and diversity assessments.
6. **Eligibility recording.** Calculate contribution weight based on the quality score and task parameters. Record the accepted proof without enabling a token transfer or claim.

Listing 1 illustrates a simplified workflow for querying the HANDB metadata table, synchronizing processed episodes from object storage, and instantiating training datasets.

4.7 Task taxonomy and verb distribution

HANVERSE organizes data along three primary axes: task, scenario (scene + object configuration), and demonstrator. The full taxonomy extends beyond the six HANVERSE-A flagship tasks of Section 4.3 to cover a structured set of categories spanning the manipulation regimes in household, logistics, and service settings. Table 5 summarizes the top-level taxonomy. Table 6 breaks out the top ten manipulation verbs within four representative categories, showing how different task domains emphasize distinct motor primitives while sharing common foundational actions (*pick*, *place*, *hold*).

Listing 1: Simplified example illustrating SQL-based episode resolution, S3 synchronization, and instantiation of training datasets.

```python
# 1. Query SQL table to resolve processed episodes
filters = {
    "robot_name": "robot_a",
    "source": "lab_a",
    "task": "task_x",
    "is_deleted": False,
}
rows = query_sql_table(filters)
# rows: [(processed_path, episode_hash), ...]

# 2. Download processed data from S3
for processed_path, episode_hash in rows:
    local_dir = f"/tmp/hanverse/{episode_hash}/"
    s3_src = f"{processed_path}/*"
    run_command(["s5cmd", "sync", s3_src, local_dir])

# 3. Instantiate dataset objects
datasets = []
for _, episode_hash in rows:
    dataset = SingleHanVerseDataset(
        root=f"/tmp/hanverse/{episode_hash}",
        mode="train",
    )
    datasets.append(dataset)

# 4. Combine datasets for training
train_dataset = MultiHanVerseDataset(datasets)
```

4.7.1 Task categories

The dataset covers a structured taxonomy of manipulation tasks spanning seven categories, summarized in Table 5. The taxonomy is designed to be jointly exhaustive across common robot manipulation regimes while remaining feasible for typical bimanual mobile manipulators.

Table 5: Task categories and representative examples. Categories span the major manipulation regimes encountered in household and logistics environments.

Category	Manipulation type	Representative tasks
Pick and Place	Single-arm grasping	Pick, place, transfer objects between containers, sort items
Fine Manipulation	Bimanual, precision	Cup-on-saucer, pour liquid, thread needle, assemble parts
Deformable Objects	Bimanual, cloth/soft	Fold clothes, bag groceries, wrap items, sheet folding
Tool Use	Extended reach, force	Sweep with broom, stir with spoon, hammer nail, use scissors
Cooking and Prep	Multi-step, sequential	Chop vegetables, scoop granular material, cook meal, prep ingredients
Cleaning and Maintenance	Surface, repetitive	Wipe surface, mop floor, scrub dishes, polish furniture
Assembly and Repair	Precision, multi-part	Assemble furniture, install hardware, connect cables, basic repairs

Within each category, the distribution of manipulation verbs provides a fine-grained view of behavioral diversity. Table 6 shows the top manipulation verbs across representative categories, illustrating how different task domains emphasize distinct motor primitives while sharing common foundational actions like *pick* and *place*.

Table 6: Top 10 manipulation verbs per category with representative frequencies. Each column is sorted independently by frequency within that category.

Logistics	Freq.	Cooking	Freq.	Cleaning	Freq.	Assembly	Freq.
pick	34,524	pick	16,197	scrub	20,320	pick	16,387
scoop	12,164	place	6,678	pick	12,297	place	5,476
place	10,322	cut	5,975	clean	7,713	adjust	5,263
fill	7,991	scoop	4,722	wipe	6,863	remove	3,305
adjust	7,802	adjust	4,496	dip	5,951	unscrew	3,222
seal	4,249	fill	2,654	place	5,480	hold	2,104
open	3,392	hold	2,153	adjust	5,225	tighten	2,052
put	3,064	slice	1,974	hold	3,358	clean	1,564
hold	3,041	remove	1,687	remove	2,862	test	1,253
insert	1,996	press	1,646	wash	2,301	put	1,196

4.8 Structured axes of diversity

The dataset is designed to capture diversity while maintaining controlled task semantics and data quality. Human data collection is organized along three primary axes:

Task and scenario diversity. Each task category includes multiple specific tasks performed across 8-12 scenes per contributing site, with 1-10 dataset units collected per scene to capture within-environment variation. Demonstrations are recorded within practical workspaces, with object positions randomized across trials. Objects are sampled from a fixed set per task within each

contributing location, while independent procurement across sites introduces substantial variation in object geometry, appearance, and material properties.

Demonstrator diversity. Data are collected from 1-8 demonstrators per contributing site. Despite identical instructions, demonstrators exhibit consistent differences in motion patterns, timing, coordination strategies, and hand trajectories. This naturally induces variation in human morphology and egocentric viewpoints due to differences in height, posture, and workspace configuration. Rather than eliminating this variation, HanVerse treats it as an inherent property of scalable human data that contributes to policy robustness.

Controlled-diversity subsets. While global aggregation provides realistic diversity, it also introduces uneven coverage across scenes and demonstrators. To enable controlled study of diversity effects, HanVerse maintains secondary datasets where data are allocated via structured assignment matrices, decoupling scene diversity and demonstrator diversity so each can be independently scaled and studied.

4.9 Task lifecycle

Tasks in HanVerse follow a defined lifecycle from creation to eligibility recording:

1. An enterprise customer or the protocol itself posts a task to the marketplace, specifying the category, required views, success criteria, and quality parameters.
2. The task enters the marketplace and becomes visible to contributors whose profiles match the task requirements (device capability, location, prior quality scores).
3. Contributors claim tasks through the HanCapture app, receive recording guidance, and submit demonstrations.
4. Submissions enter the HanDB processing pipeline for pose estimation, quality scoring, and provenance verification.
5. Accepted submissions update proof and eligibility state based on the quality score $Q(\mathcal{E}_i)$ and the task's configured weight.
6. Processed data enters the enterprise-accessible corpus, available through dataset downloads, streaming subscriptions, or API queries.

5 Quality Assurance and Security

A data marketplace must price episodes by utility. HanVerse defines a composite quality score $Q(\mathcal{E}) \in [0, 1]$ that determines both contributor compensation and dataset composition.

5.1 HanScore: quality decomposition

Let $\{V_k\}_{k=1}^{K}$ be a set of automated validators producing bounded scores across dimensions including: timing integrity, pose estimation confidence, visual clarity, task completion evidence, and metadata completeness. The base quality score is:

$$Q_0(\mathcal{E}) = \sum_{k=1}^{K} w_k \cdot V_k(\mathcal{E}), \quad \sum_k w_k = 1. \tag{3}$$

To discourage spam and near-duplicates, the system applies a novelty factor $N(\mathcal{E})$ based on embedding distance in a visual representation space. Let Φ denote a frozen visual encoder (e.g., DINOv3) and $\mathcal{D}$ the existing corpus:

$$d_{\min}(\mathcal{E}) := \min_{\mathcal{E}' \in \mathcal{D}} \|\Phi(\mathcal{E}) - \Phi(\mathcal{E}')\|_2. \tag{4}$$

A monotone, saturating map ensures near-duplicates receive near-zero novelty while genuinely novel episodes saturate toward 1:

$$N(\mathcal{E}) = \begin{cases} 1, & \text{if } \mathcal{D} = \emptyset, \\ 1 - \exp\big(-\alpha\, d_{\min}(\mathcal{E})\big), & \text{otherwise,} \end{cases} \tag{5}$$

where $\alpha > 0$ is a scale parameter calibrated to the embedding geometry.

Finally, the diversity value $\Delta(\mathcal{E})$ captures the geographic, demographic, and environmental novelty of the contributor and scene relative to the existing dataset distribution. The composite quality score is:

$$Q(\mathcal{E}) = \mathrm{clip}_{[0,1]}\big(\lambda_0 Q_0(\mathcal{E}) + \lambda_1 N(\mathcal{E}) + \lambda_2 \Delta(\mathcal{E})\big), \quad \lambda_i \geq 0,\ \sum_i \lambda_i = 1. \tag{6}$$

5.2 veHAN review network

Experienced contributors may participate in the veHAN review network by locking **$HAN** tokens after the lock vault is live. Reviewers focus on submissions that pass automated screening but require human judgment (ambiguous task completion, borderline quality, novel task types). In the current framework, veHAN is non-transferable participation power. It can influence future protocol eligibility, but it does not create a claimable reward, APY, revenue share, or payout right at TGE.

The locking requirement creates commitment: participants who lock longer signal durable alignment with dataset quality and protocol health. Any future review rewards, slashing, or fee participation would require separate governance, contracts, and legal review. The launch-safe objective is narrower: use veHAN to activate proof-gated participation state without enabling claims.

5.3 Anti-sybil mechanisms

Sybil attacks (one person operating many accounts to farm eligibility) are mitigated through multiple complementary mechanisms:

Device attestation. Each capture device registers a device-bound key through a decentralized identifier (DID) model. Hardware attestation, where available, strengthens the binding between key and physical device. A single device can only be registered to one contributor account.

Behavioral fingerprinting. Contribution patterns (timing, location, motion characteristics, hand morphology) create a behavioral signature that is difficult to replicate across synthetic accounts.

Diversity-weighted eligibility. The diversity multiplier $\alpha(\mathrm{div}_i)$ in Eq. (1) naturally penalizes Sybil strategies: one person cannot fake being in 50 countries with 50 different hand morphologies. Geographic and demographic novelty must be genuine to receive diversity weight.

5.4 Threat model

Table 7 summarizes the primary threats and corresponding mitigations.

Table 7: Threat model with detection mechanisms and economic disincentives.

Threat	Detection	Disincentive
Fake video (deepfake, screen recording)	Cross-modal consistency (IMU vs. visual motion), device attestation, temporal analysis	Zero eligibility + account suspension; reviewer reputation reduced if fake data is approved
Sybil accounts	Device binding, behavioral fingerprinting, location analysis	Diversity multiplier penalizes synthetic diversity; eligibility per fake account is below cost of maintaining it
Near-duplicate spam	Embedding-space novelty scoring, deduplication pipeline	Near-zero novelty score ⇒ near-zero eligibility weight
Reviewer collusion	Random audit sampling, cross-reviewer consistency checks, delayed eligibility finalization	Reputation score reduction; future review access gated
Data poisoning	Automated quality prescreening, downstream policy evaluation, community flagging	Contributor reputation permanently damaged; future submissions require higher validation thresholds

6 Hydra-Gen: Motion World Foundation Model

Hydra-Gen is the generative motion layer of the Humanoid Network: a kinematic motion diffusion model trained on hundreds of hours of optical motion capture that produces controllable humanoid motion from a text prompt and a flexible suite of kinematic constraints. The model is trained to satisfy three jointly demanding criteria: (i) motion quality on par with optical mocap, (ii) a versatile and directable interface with intuitive controls, and (iii) the ability to produce a diverse corpus of behaviours when used as an autonomous data generator for downstream training pipelines.

6.1 Controllability suite

Hydra-Gen supports five classes of kinematic constraint, combinable arbitrarily within a single generation:

1. **Text prompt.** Natural-language description of the target motion. Inputs expect a subject prefix ("A person…", "An old person…", "A zombie…"). Multi-prompt chaining produces long sequences stitched together in the demo interface.
2. **Full-body keyframes.** Sparse pose keyframes at specified times constrain the full body pose at those frames. Useful for in-betweening between mocap clips.
3. **End-effector keyframes.** Sparse position and/or rotation keyframes for hands or feet. Useful for object-interaction constraints and reach targets.
4. **2D root waypoints.** Sparse planar position or heading constraints on the root across the trajectory. Useful for navigation-mesh-driven paths.
5. **Dense 2D paths.** A continuous planar curve the root must follow. Useful for locomotion constraint randomization during bulk dataset synthesis.

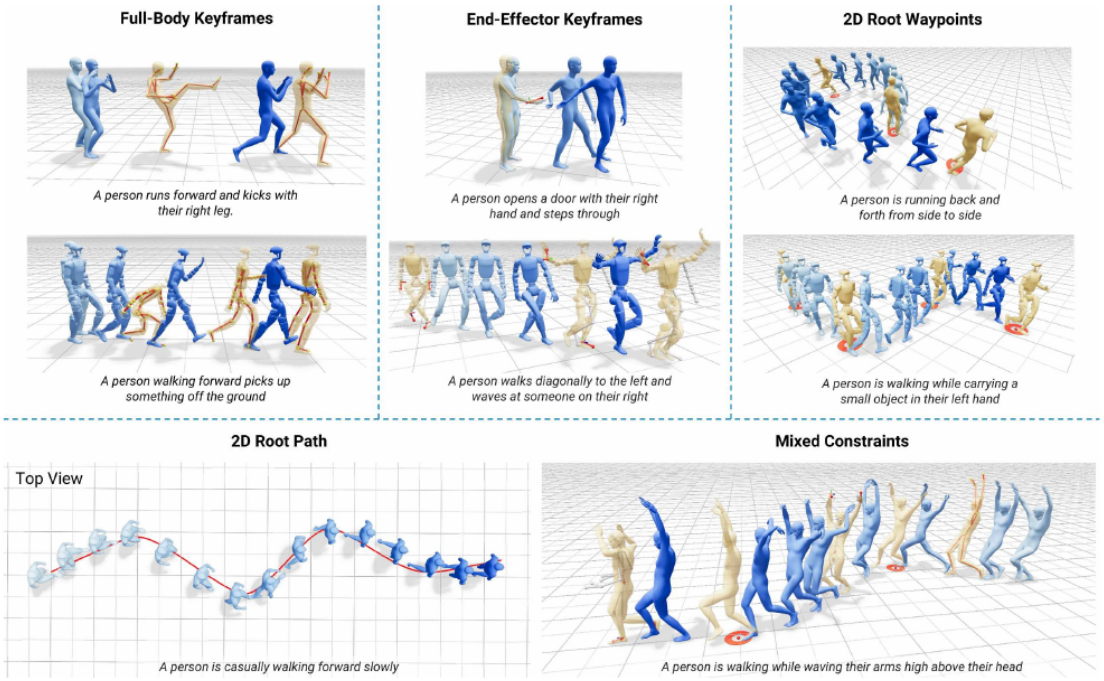


Figure 5: HYDRA-Gen controllable motion generation. The model supports flexible and intuitive control through text prompting combined with an extensive suite of kinematic constraints. Constrained joints are shown in red; generated poses at constrained frames are highlighted in yellow. Time progression is indicated by lighter to darker blue. From left to right, top to bottom: full-body keyframes, end-effector keyframes, 2D root waypoints, 2D root path, and mixed constraints.

On an RTX 3090, generation takes 2–5 seconds depending on sequence duration (max 10 seconds per generation). On the HANVERSE-A test suite the model achieves 3.21 cm mean full-body keyframe error, 3.63 cm end-effector position error, 6.88° end-effector rotation error, and 3.63 cm root position error. Light post-processing (foot-locking plus a short inverse-kinematics pass) brings generated outputs to exact constraint satisfaction when strict hitting is required; the experimental numbers in Section 6.6 do *not* apply this post-processing, so the reported errors reflect the model alone.

6.2 Motion representation

Each pose frame is represented as a feature vector $[r_p, r_a, j_p, j_v, j_a, f]$:

- $r_p \in \mathbb{R}^3$ — smoothed global root position. The planar pelvis trajectory is heavily smoothed; the vertical (height) channel is kept unchanged. Smoothing replaces the raw hip sway with a path resembling the curve an animator would draw by hand, and provides a stable frame of reference for joint-position encoding.
- $r_a \in \mathbb{R}^2$ — heading direction as $[\cos\psi, \sin\psi]$, derived from the projection of the up-vector cross hip-vector into the ground plane. The sin/cos encoding avoids heading discontinuities.
- $j_p \in \mathbb{R}^{3J}$ — global joint positions. Planar coordinates are taken relative to the smoothed root; the vertical is global. Joints are *not* rotated into a root-relative heading frame — retaining the world-frame heading is what allows motions like somersaults and cartwheels where the heading is ill-defined.
- $j_v \in \mathbb{R}^{3J}$ — global joint velocities computed by finite differences on j_p.

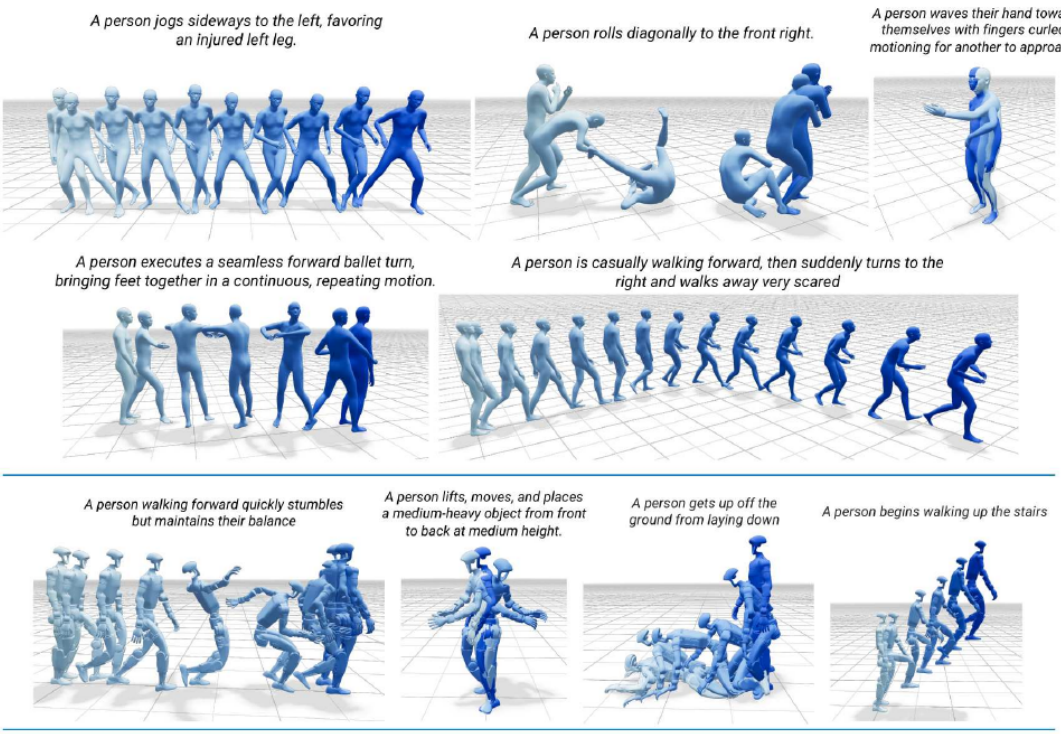


Figure 6: HYDRA-Gen qualitative text-to-motion results. (Top) Diverse high-quality motions generated on the SOMA body skeleton from short text prompts. Time progression is indicated by lighter to darker blue. (Middle) The same motions can be retargeted to the Unitree G1 robot skeleton at generation time, producing demonstration data directly on the target embodiment. (Bottom) Ten different generations of the same prompt ("A person puts an object down leftward") demonstrate the diversity of HYDRA-Gen samples.

- $j_a \in \mathbb{R}^{6J}$ — global joint orientations in the 6D rotation representation [17]. Using *global* (rather than local / chained) rotations enables sparse rotation constraints that do not require running a forward-kinematics chain.
- $f \in \{0,1\}^4$ — binary foot-contact flags for [left heel, left toe, right heel, right toe].

The feature vector decomposes naturally into a root component $r_{\text{glob}} = [r_p, r_a]$ and a body component $b = [j_p, j_v, j_a, f]$. This decomposition is the empirical foundation of the two-stage denoiser architecture below.

6.3 Two-stage transformer denoiser

HYDRA-Gen uses a two-stage transformer denoiser architecture, illustrated in Fig. 8. Motion synthesis is framed as a diffusion problem: the forward process applies Gaussian noise to a clean motion sequence x_0 over $T = 1000$ timesteps; the reverse process is learned as a denoiser $D_\theta(x_t, C, t)$ that predicts clean motion $\hat{x}_0$ given a noisy input x_t, conditioning C, and timestep t.

Constraint imputation. Constraints are handed to the model as partial pose features x_{tgt} plus a binary mask m. The imputed input is $\tilde{x}_t = m \odot x_{\text{tgt}} + (1-m) \odot x_t$, concatenated with the mask itself ($x_{\text{in}} = [\tilde{x}_t; m]$) before tokenization. This lets the model treat constraints as exact overwrites rather than soft targets, which is what produces the sub-5-cm constraint accuracy reported in Section 6.1.

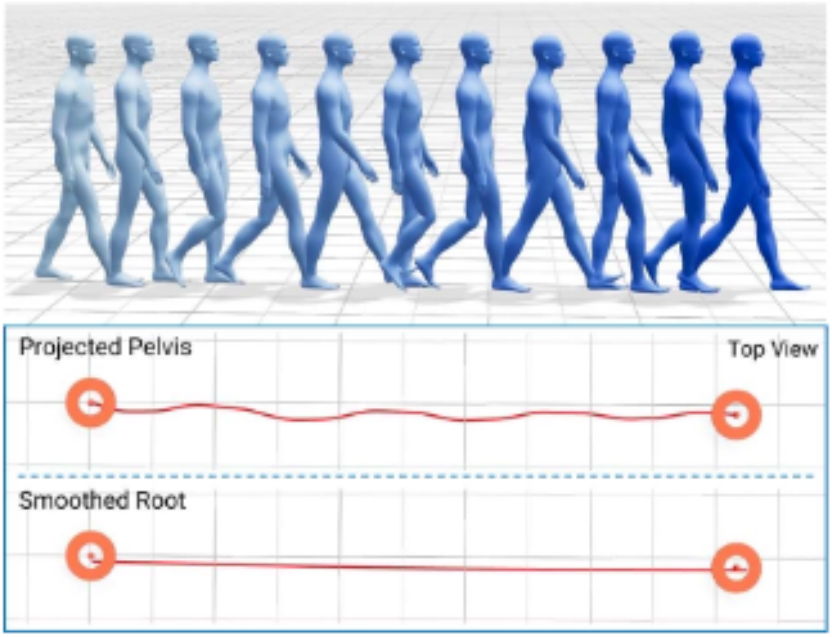


Figure 7: Smoothed root representation. For a simple walking motion, the projected pelvis path (top trace) captures the sway of the hips, while the smoothed root (bottom trace) is nearly a straight line. The smoothed trajectory provides a stable frame of reference for encoding joint positions and emulates the curves drawn by hand in practical animation tools; it is critical to HYDRA-Gen's constraint-tracking accuracy.

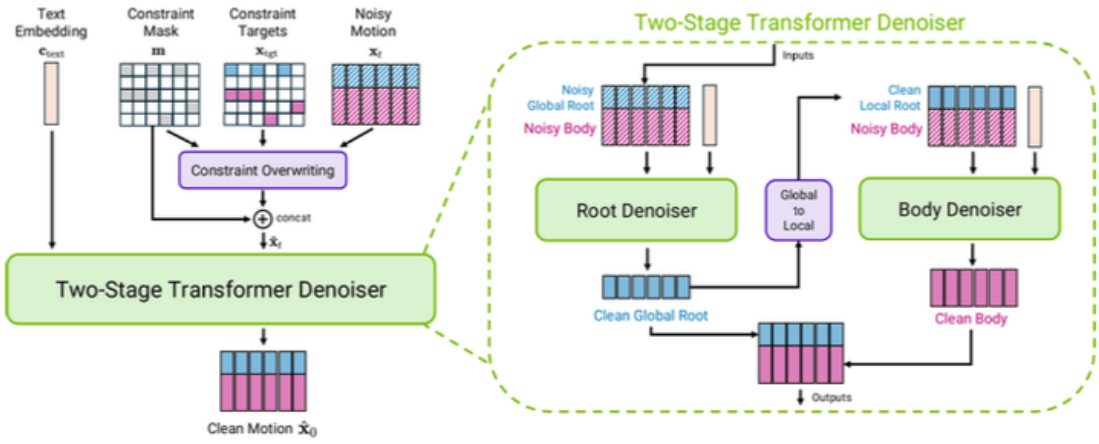


Figure 8: HYDRA-Gen two-stage transformer denoiser. (Left) Constraints are specified as partial pose features x_{tgt} plus a binary mask m; they are imputed into the noisy motion before denoising begins so the model always sees the constraints as an overwrite on the noisy input. (Right) The denoiser decomposes into a root denoiser that produces global root motion and a body denoiser that consumes root-conditioned local features to produce body motion. Both stages run at every denoising step.

Conditioning. Three conditioning streams are appended as additional tokens: a 4096-dim text embedding c_{text} (produced by an LLM-based sentence encoder; chosen over CLIP and T5 after early experiments), a 2-dim desired first-frame heading c_{dir}, and a block of $P = 49$ extra register-

like conditioning tokens c_{extra}. All tokens are embedded to the same dimensionality; sinusoidal positional encoding is added.

Two-stage decomposition. The root denoiser D_{root} consumes the full noisy input x_{in} and produces the predicted global root motion $\hat{r}_{\text{glob},0}$. The predicted global root is then converted into a local-frame representation $r_{\text{local}} = [\dot{r}_a, \dot{r}_{pxz}, r_{py}]$ (angular velocity of heading, planar translation velocity, absolute height), concatenated with the body features from x_{in}, and passed to the body denoiser D_{body} which outputs the predicted body motion $\hat{b}_0$. The final output $\hat{x}_0 = [\hat{r}_{\text{glob},0}; \hat{b}_0]$ is the concatenation of the two stages. Unlike prior approaches that run root denoising fully before body denoising, HYDRA-Gen runs both stages interleaved at every denoising step.

Capacity. Each transformer is 16 layers, 8 heads, latent dim 1024, for a total of 282M parameters across the two stages.

6.4 Training curriculum

Training follows the DDPM framework with a modified simplified loss. Each iteration samples a clean motion x_0, a diffusion timestep $t \sim \text{Uniform}\{1..T\}$, and a Gaussian noise vector ε, and forms the noisy x_t. The training loss is:

$$\begin{aligned}\mathcal{L}_{\text{gen}} = \gamma_1 \|\hat{r}_{p,0} - r_{p,0}\|_1 + \gamma_2 \|\hat{r}_{a,0} - r_{a,0}\|_1 \\ + \gamma_3 \|\hat{j}_{p,0} - j_{p,0}\|_1 + \gamma_4 \|\hat{j}_{v,0} - j_{v,0}\|_1 + \gamma_5 \|\hat{j}_{a,0} - j_{a,0}\|_1 \\ + \gamma_6 \|\hat{f}_0 - f_0\|_1 + \gamma_7 \|\text{FK}(\hat{j}_{a,0}) - j_{p,0}\|_1,\end{aligned} \tag{7}$$

where $\|\cdot\|_1$ is smooth L1 (quadratic near zero, linear far from zero), and $\text{FK}(\cdot)$ is the forward-kinematics operator that maps predicted joint rotations to their implied joint positions — a consistency regularizer between predicted rotations and predicted positions. Loss weights are $\gamma_1 = \gamma_3 = \gamma_5 = 10.0$ (root pos, joint pos, joint rot), $\gamma_2 = 2.0$ (root heading), $\gamma_4 = 3.0$ (joint velocity), $\gamma_6 = 4.0$ (foot contact), $\gamma_7 = 5.0$ (FK consistency). Variable-length sequences within a batch are masked accordingly.

Curriculum. Training runs in two phases of 500,000 steps each (1M steps total). Phase 1 is pure text-to-motion with no kinematic constraints and dropout 0.1. Phase 2 adds mixed kinematic constraints (full-body keyframes, end-effector keyframes, 2D root paths, foot-contact configurations) sampled from a curriculum that ramps up the maximum number of keyframes linearly from 1 to 20. 25% of phase 2 steps mix two constraint patterns together; 10% of steps use no constraints (text-only). Dropout is removed in phase 2 to avoid dropping out constraint overwrites. Text conditioning is dropped 10% of the time (classifier-free guidance). The optimizer is Adam-atan2 with learning rate 2×10^{-5}; EMA is applied every 10 steps with decay 0.995.

Best configuration. The full 282M-parameter model is trained at batch size 2048 across 16 NVIDIA A100 (SXM4-80GB) GPUs at 30 fps generation. Max sequence length is 10 seconds. First-frame heading is randomized per batch to break spurious heading dependencies.

6.5 Inference and classifier-free guidance

HYDRA-Gen inference uses DDIM with 100 denoising steps by default. A decomposed classifier-free guidance formula combines the unconditional prediction $D_\emptyset$ with text- and constraint-conditional predictions:

$$\hat{x}_0 = D_\emptyset + w_{\text{text}}(D_{\text{text}} - D_\emptyset) + w_{\text{constr}}(D_{\text{constr}} - D_\emptyset), \tag{8}$$

with default guidance weights $w_{\text{text}} = 2$ and $w_{\text{constr}} = 2$. Test-time gradient guidance was explored and not retained — it produced minimal quality improvement, substantially increased generation time, and was numerically unstable.

Multi-prompt sequencing. Longer motion sequences are produced by generating each prompt in sequence with full-body keyframe constraints in an overlap region between consecutive prompts. A short post-generation blending pass smooths the shared frames.

Motion post-processing. For authoring or bulk-dataset use, a lightweight post-processing pass applies foot-locking and inverse-kinematic cleanup using the predicted foot-contact flags, and runs a short optimization to ensure exact constraint hitting. Post-processing is disabled during the experimental evaluation in Section 6.6 for fair comparison.

6.6 Benchmarks: ablations and scaling

HYDRA-Gen is evaluated on a held-out 10% split of the training corpus, split by unique behaviours so the test set contains novel action types. The evaluation subset is approximately 5,000 motions. Metrics include top-3 retrieval precision (R@3) and Fréchet Inception Distance (FID) under a motion-text retrieval model, foot-skate during frames predicted in static contact, foot-contact classification accuracy, full-body position error, end-effector position and rotation errors, and 2D root position error. FID is multiplied by 100 for readability.

Table 8: HYDRA-Gen ablation study on the held-out test set. All ablations trained at 20 fps with medium batch (8 GPU); the full Sec. 6.1 configuration is 30 fps / 16 GPU. Each ablation hurts a specific axis, confirming the role of each design choice.

Method	**Overview prompt**				**Fine-grained prompt**			
	R@3↑	FID↓	Skate↓	Contact↑	R@3↑	FID↓	Skate↓	Contact↑
Ground truth	75.6	0.00	2.21	1.00	79.4	0.00	2.23	1.00
Full model	**71.9**	**1.85**	**3.87**	0.98	63.5	**1.67**	**3.88**	0.98
One-stage arch	71.5	1.65	7.59	0.94	63.5	1.51	6.80	0.95
Second-stage global	70.3	1.87	4.17	0.98	63.2	1.66	4.07	0.98
No smoothed root	71.6	1.75	4.39	0.97	**64.0**	1.55	4.27	0.98
No extra tokens	70.9	1.95	4.28	0.97	61.6	1.77	4.17	0.98
No train curriculum	71.3	1.84	3.92	0.98	63.2	1.66	3.91	0.98

Table 8 summarises the ablation study. Removing the two-stage decomposition ("One-stage arch") roughly doubles foot-skate (3.87 → 7.59 cm/s) and triples full-body position error relative to the full model — confirming that body-conditioned-on-root prediction is far easier than joint denoising of the combined feature vector. Removing the smoothed-root representation ("No smoothed root") increases foot-skate (3.87 → 4.39 cm/s) and qualitatively degrades straight-line walking toward a stealthy / old-person style. Removing the training curriculum ("No train curriculum") preserves text-following metrics but doubles full-body position error (2.67 → 5.80 cm for constrained full-body keyframes) — showing that the second-phase constraint curriculum is what produces constraint accuracy, while first-phase text-only training produces the baseline motion quality. Each ablation hurts a distinct axis; no single component dominates.

Table 9 summarises the scaling analysis. Dataset size primarily affects constraint accuracy and foot-skate (because the retained held-out split preserves behaviour diversity, the retrieval-based R@3 and FID metrics are relatively insensitive to training-set size). Model size scales every metric roughly monotonically from 56M to 282M parameters; further scaling beyond 500M–1B raises training-stability challenges and likely requires additional data. Batch size scales modestly across the board, with the largest configuration (16 GPU / batch 2048) producing the best retrieval (R@3 = 73.6) and best constrained pose accuracy (2.33 cm full-body). Figure 9 visualises the three scaling axes.

Table 9: HYDRA-Gen scaling analysis. (Top) Dataset size primarily drives constraint accuracy; text-following metrics are relatively insensitive to subsets because the eval set preserves behaviour diversity. (Middle) Model size scales all metrics. (Bottom) Batch size (via GPU count) improves everything.

Configuration	R@3↑ (overview)	FID↓ (overview)	Full-body pos (cm)↓	End-eff pos (cm)↓	End-eff rot (deg)↓	2D root pos (cm)↓
Ground truth	75.6	0.00	—	—	—	—
Data size						
Full dataset	71.5	1.84	2.77	3.31	5.36	3.29
50% dataset	70.8	1.81	3.13	3.56	6.29	3.32
10% dataset	71.0	2.07	4.60	6.91	10.03	4.83
Model size						
Large (282M)	71.9	1.85	2.67	3.09	4.18	2.90
Medium (148M)	69.2	2.36	3.26	3.72	4.70	3.34
Small (56M)	64.0	3.10	3.56	3.98	11.27	3.49
Batch / GPUs						
Large (16 GPU, batch 2048)	73.6	1.61	2.33	2.71	4.09	2.35
Medium (8 GPU, batch 1024)	71.9	1.85	2.67	3.09	4.18	2.90
Small (4 GPU, batch 512)	69.4	2.01	2.97	3.68	5.61	3.42

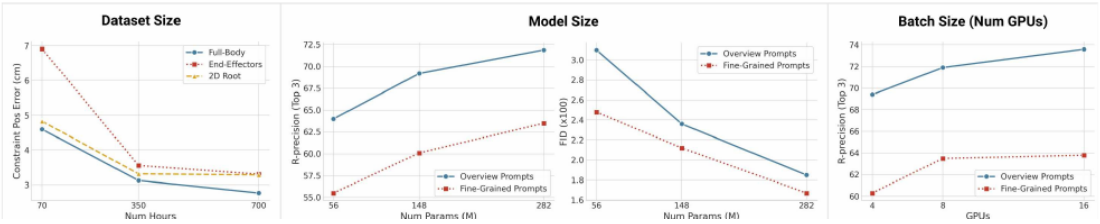


Figure 9: HYDRA-Gen scaling. Scaling dataset size (left), model size (centre), and batch size / GPU count (right) improves both controllability and motion quality. Increased dataset size yields the largest gains in constraint following; model and batch size scaling disproportionately help text-following (R-precision) and motion quality (FID).

Figure 10: HYDRA-Gen as a robot demonstration generator. HYDRA-Gen motions generated directly on the Unitree G1 skeleton ("Generated") are tracked by HYDRA-Track on a simulated humanoid ("Simulated") and deployed zero-shot to the physical robot ("Deployed"). Prompts: "a person does a dance stepping back and forth" (left triptych), "a person does jumping jacks" (right triptych).

7 Hydra-Track: Physics Validation and Universal Control

The HYDRA-Gen outputs of Section 6 are kinematically plausible — by construction, the retrieval and FID metrics confirm they match the corpus distribution — but *kinematic plausibility is not physical executability*. A generated motion is useful for robot policy training only if a physical robot can actually follow it without falling, slipping, or losing balance. HYDRA-Track is the layer of the Humanoid Network that answers this question. Every generated motion destined for reward eligibility is executed on a simulated humanoid by a universal tracking policy; the tracking metrics returned by the policy are the gate between contribution and eligibility.

HYDRA-Track is a reinforcement-learning control policy trained with proximal policy optimization [18] to track retargeted human motion on the Unitree G1 humanoid [19]. Relative to prior motion-imitation work [20, 21] it supersizes scale along three axes: network capacity (1.2M → 42M parameters), data (100M frames spanning 700 hours of motion), and compute (9,000 GPU hours). The design target is a single-policy system that handles robot motion, human motion, and hybrid keypoint motion through a shared latent representation, enabling one tracker to be driven by multiple heterogeneous motion sources (generated motions from HYDRA-Gen, teleoperation from VR, video-estimated motion, VLA policies) without retraining.

7.1 Universal control policy architecture

HYDRA-Track is a Markov decision process $M = \langle S, A, T, R, \gamma \rangle$ solved by PPO with discount $\gamma = 0.99$. The state decomposes into a proprioceptive component $s_t^p = (q_t, \dot{q}_t, \omega_t, \psi_t, a_{t-1})$ (joint pose, joint velocity, root angular velocity, gravity in root frame, previous action) and a motion-command component s_t^g. The motion-command component comes in three forms: a robot motion command g_r, a human motion command g_h (3D SMPL-X joint positions [22]), or a hybrid command g_m (sparse upper-body keypoints at the current frame plus lower-body robot motion over a lookahead window). All quantities are expressed in the robot's local heading frame for rotational invariance.

The action space is 29-dimensional target joint positions tracked by on-robot per-joint PD controllers. Rewards decompose into tracking terms (exponential distance on root orientation, body-link position, body-link orientation, body-link linear velocity, body-link angular velocity, all in the root frame except root orientation in the world frame) and penalty terms (action-rate penalty, joint-limit violation penalty, undesired-contact penalty excluding ankles and wrists). The reward schedule is summarized in Table 10.

Table 10: HYDRA-Track reward design. Orientations are represented as 6D rotations [17]. Superscript g: goal (target) state from s_t^g; superscript p: proprioceptive (current) state from s_t^p; B: set of tracked body links; "rel.": quantities in root frame.

Reward term	**Form**	**Weight**
Tracking rewards $R(s_t^p, s_t^g)$		
Root orientation	$\exp(-\|o^p_{t,r} - o^g_{t,r}\|_2^2 / 0.4^2)$	0.5
Body-link position (rel.)	$\exp(-\|\Delta p^{\text{rel}}\|_2^2 / 0.3^2)$	1.0
Body-link orientation (rel.)	$\exp(-\|\Delta o^{\text{rel}}\|_2^2 / 0.4^2)$	1.0
Body-link linear velocity	$\exp(-\|\Delta v\|_2^2 / 1.0^2)$	1.0
Body-link angular velocity	$\exp(-\|\Delta \omega\|_2^2 / 3.14^2)$	1.0
Penalty terms $P(s_t^p, a_t)$		
Action rate	$\|a_t - a_{t-1}\|_2^2$	-0.1
Joint-limit violation	$\sum_j \mathbf{1}[q_{t,j} \notin [q_{t,j}^{\min}, q_{t,j}^{\max}]]$	-10.0
Undesired contact	$\sum_{c \notin \{\text{ankles, wrists}\}} \mathbf{1}[\|F_c\| > 1.0\,\text{N}]$	-0.1

Three specialized MLP encoders (E_r for robot, E_h for human SMPL-X, E_m for hybrid keypoints) all with hidden dimensions $[2048, 1024, 512, 512]$ map their respective motion commands into a shared latent. A finite scalar quantizer [23] discretizes the latent into a universal motion token $z \in \mathcal{Z}$ (with D_z dimensions and L_z levels per dimension). Two decoders consume the token: a robot control decoder D_c (hidden $[2048, 2048, 1024, 1024, 512, 512]$) produces the action Gaussian, and an auxiliary robot-motion decoder D_r reconstructs the robot motion command — acting as an implicit human-to-robot retargeter when the input is a human motion.

The training objective combines four losses:

$$\mathcal{L}_{\text{track}} = \mathcal{L}_{\text{PPO}} + \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{token}} + \mathcal{L}_{\text{cycle}}, \tag{9}$$

where $\mathcal{L}_{\text{recon}} = \|D_r(z_r) - g_r\|^2 + \|D_r(z_h) - g_r\|^2 + \|D_r(z_m) - g_r\|^2$ enforces that the decoder reconstruct the robot motion command from any of the three token variants; $\mathcal{L}_{\text{token}} = \|z_r - z_h\|^2$ forces cross-embodiment alignment in the latent; and $\mathcal{L}_{\text{cycle}} = \|E_r(D_r(z_h)) - z_r\|^2$ enforces cycle consistency when a human-encoded token is decoded to a robot motion and re-encoded.

Domain randomization. Training applies uniform-distribution randomization across static and dynamic friction coefficients, restitution, default joint positions, base centre-of-mass offsets, external root linear and angular velocity pushes, and target-motion jitter (position, orientation, linear velocity, angular velocity, joint angles). Representative parameters: static friction $\mu_s \sim U[0.3, 1.6]$; dynamic friction $\mu_d \sim U[0.3, 1.2]$; restitution $e \sim U[0, 0.5]$; root linear velocity perturbation $v_{x,y} \sim U[-0.5, 0.5]$ m/s over push durations $\Delta t \sim U[1, 3]$ s.

Adaptive motion sampling. The training corpus is partitioned into 1-second bins. Each bin tracks a running failure rate f_i (capped at $\beta \bar{f}$ with $\beta = 200$). Preliminary sampling weight is $\hat{p}_i \propto f_i$; final probability blends with uniform: $p_i = \alpha \hat{p}_i + (1 - \alpha) \cdot 1/N$ with $\alpha = 0.1$. The scheme concentrates training effort on difficult motions without abandoning coverage of the full corpus.

7.2 Tracking performance and scaling

HYDRA-Track is evaluated on a uniformly random subset of retargeted AMASS [24], 9 hours / 1,602 trajectories held out during training. Evaluation uses Isaac Lab [25]; baselines are reproduced in MuJoCo [26] for fairness. Metrics: success rate (full trajectory tracked to completion), MPJPE (E_{mpjpe}, mm), velocity error (E_{vel}, mm/frame), acceleration error (E_{acc}, mm/frame^2). The termination criterion during evaluation is a root-height deviation > 0.25 m or root-orientation deviation > 1 radian from the reference. The headline comparison against tracking baselines is shown in Table 11.

Table 11: HYDRA-Track tracking performance on held-out retargeted AMASS (Unitree G1). Accuracy columns (MPJPE, E_{vel}, E_{acc}) are evaluated on trajectories that tracked to completion; failures are excluded from the accuracy columns but counted in the success-rate column.

Method	**Succ. rate (%)**	**MPJPE (mm)**	**E_{vel} (mm/fr)**	**E_{acc} (mm/fr^2)**
Any2Track	58.3	202.3	14.9	6.1
GMT	84.2	65.0	12.7	4.9
BeyondMimic	94.3	57.4	9.1	2.5
Hydra-Track (sim)	**99.6**	**42.7**	**4.1**	**1.2**
HYDRA-Track (real, 50 trajectories)	**100.0**	40.9	—	—

The scaling behaviour across the three supersizing axes is summarized below. **Data scaling:** with model size and compute fixed, training on 0.4M, 7.4M, and 100M frames drives MPJPE from roughly 49 to 38 to 37 mm and success from 93.5% to 97.5% to 98%. **Model scaling:** at fixed data and compute, moving from 1.2M to 16M to 42M parameters improves all metrics monotonically. **Compute scaling:** at fixed data and model, moving from 500 to 2,000 to 9,000 GPU hours also improves all metrics monotonically. Data is the single largest driver; parallelism matters independently of data size (an 8-GPU run reaches a worse asymptote than a 128-GPU run even with the same data).

Real-world transfer is evaluated by executing the trained policy on 50 diverse real trajectories on a physical Unitree G1 (dance, jumps, loco-manipulation). All 50 trajectories complete without

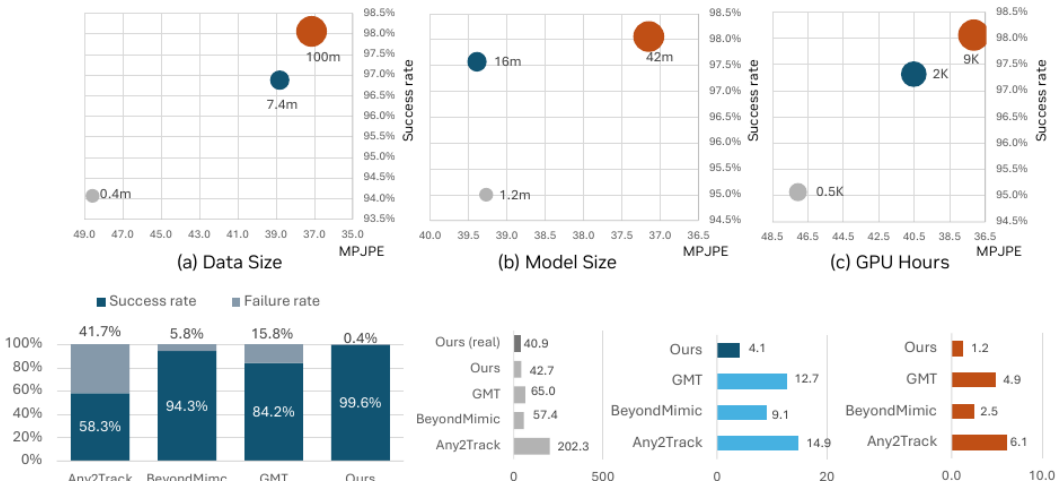


Figure 11: HYDRA-Track scaling and baseline comparison. (Top) MPJPE (motion imitation error) as a function of dataset size, model size, and compute — all three axes drive steady improvement. (Bottom) Comparison against tracking baselines on held-out retargeted AMASS: HYDRA-Track achieves higher success rate and lower tracking, velocity, and acceleration errors than Any2Track, GMT, and BeyondMimic.

failure — 100% zero-shot sim-to-real success. Mean per-joint position error during execution is 40.9 mm, slightly below the simulated value due to the domain randomization margin.

7.3 Metrics surfaced to the eligibility gate

For every submitted motion $\mathcal{E}$, HYDRA-Track reports the tuple (s, d, μ, ϕ) to the eligibility-gate logic:

- **Tracking survival** $s(\mathcal{E}) \in \{0, 1\}$ — whether the policy completed the full reference trajectory without triggering the fall termination criterion (> 0.25 m root-height deviation or > 1 rad root-orientation deviation).
- **Tracked duration** $d(\mathcal{E})$, in seconds — for trajectories that failed, how long the policy tracked before falling. Falls within one second are treated as spam; failures between one and two seconds receive a reduced band.
- **MPJPE** $\mu(\mathcal{E})$, in millimetres — the mean per-joint position error during the tracked frames.
- **Foot-skate** $\phi(\mathcal{E})$, in cm/s — the mean speed of the stance foot during frames in which the foot-contact flag indicates static contact. Foot-skate above a spam threshold is treated as a physics violation independent of the survival flag.

These four metrics feed the bucketed eligibility schedule of Section 5. The top band (low MPJPE, low foot-skate, full tracking survival) receives the top multiplier; the bottom band (survival fail, fall within 1 s) is rejected or flagged.

7.4 Sim-to-real deployment and foundation-model integration

HYDRA-Track is deployed onboard the Unitree G1 via Jetson Orin with TensorRT and CUDA Graph acceleration. The system runs the policy at 50 Hz and streams joint targets to the low-level controller at 500 Hz via the "latest-data-wins" API. User input is accepted at 100 Hz. A generative

kinematic motion planner produces 0.8–2.4 second motion segments at 10 Hz that are blended into the control objective. Policy forward-pass is 1–2 ms; kinematic planner forward-pass is 12 ms.

The universal token abstraction enables direct integration with upstream foundation models. A vision-language-action policy (fine-tuned GR00T N1.5 [27]) is used as an external "System 2" planner that emits teleoperation-format commands (head and wrist SE(3) poses, base height, navigation command); these commands are consumed by HYDRA-Track's hybrid encoder and executed as "System 1" reactive control. On an apple-to-plate bimanual pick-and-place task, fine-tuning the VLA on 300 VR-teleoperated trajectories reaches 95% success over 20 trials — demonstrating that a high-level planner can drive the whole-body tracker without modality-specific retraining.

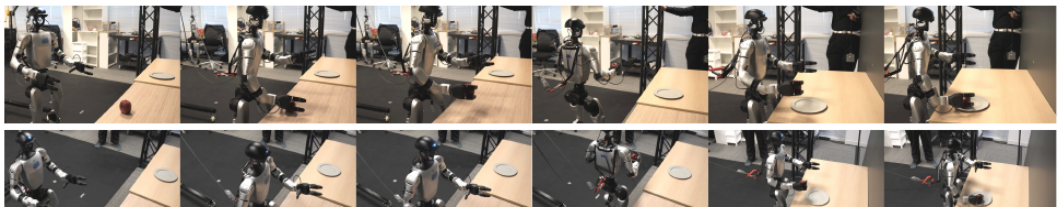


Figure 12: Apple-to-plate bimanual mobile manipulation on the Unitree G1. The sequence shows the VLA-controlled physical robot completing a pick-and-place task. The VLA emits teleoperation-format commands (head and wrist SE(3) poses, base height, navigation command) which are executed by HYDRA-Track. The model is fine-tuned on 300 VR-teleoperated trajectories and reaches 95% success over 20 trials, demonstrating that foundation-model-driven task planning composes cleanly with the universal tracking policy.

8 Proof Attestation: the Oracle Layer

The ORACLE layer is the bridge between off-chain computation (data ingestion, quality scoring, physics validation) and protocol records (provenance, eligibility state, and enterprise verification). Its job is to produce tamper-evident records of every proofed contribution: what was submitted, who submitted it, what score and physics metrics it received, and whether it passed the relevant quality gates.

8.1 Attestation structure

For each contribution $\mathcal{E}_i$ that passes quality and physics gates, the ORACLE constructs a signed attestation A_i of the form:

$$A_i = \sigma_{\mathsf{sk}_{\text{ORACLE}}}\big(\text{hash}(\mathcal{E}_i) \,\|\, Q(\mathcal{E}_i) \,\|\, (s_i, d_i, \mu_i, \phi_i) \,\|\, \text{id}_i \,\|\, t_i\big), \tag{10}$$

where σ denotes a signature under the ORACLE's private key, $\text{hash}(\mathcal{E}_i)$ is the content-addressed episode hash from Section 4.2.1, $Q(\mathcal{E}_i)$ is the composite quality score, $(s_i, d_i, \mu_i, \phi_i)$ are the physics metrics from Section 7.3, id_i is the contributor identifier, and t_i is a Unix timestamp. The signature follows typed-data signing conventions for user-readable verification in wallets.

8.2 Eligibility record flow

Attestations flow into the eligibility layer. Accepted, proofed motions can update non-claimable contribution state, future protocol reward weight, and enterprise provenance records. The record is designed to be auditable without minting tokens, transferring reserves, or enabling reward claims.

This design lets the network rehearse TGE mechanics cleanly while the canonical HAN token is created externally and while Humanoid Network keeps reward execution disabled. The same cryptographic provenance record can support enterprise verification, future governance, and later claim-controller activation if approved.

8.3 Adversarial model

The ORACLE is a trust-assumption surface: a compromised ORACLE key could sign attestations for fictitious contributions. Mitigations include key management using threshold signatures across multiple protocol participants, publishing the daily aggregate of attestations in an append-only public log that any auditor can reconcile against protocol records, and restricting any future high-value claim activation to a multi-signature or governance-controlled contract. These mechanisms do not eliminate trust in the ORACLE, but they bound the damage a single-key compromise can do and create detectable evidence of drift between off-chain records and on-chain state.

9 Human-to-Robot Transfer

HANVERSE does not aim to invent new robot learning algorithms. The research community is producing increasingly capable cross-embodiment models, and HANVERSE's role is to supply the diverse, high-quality human data these models require. This section describes how HANVERSE data integrates with established human-to-robot transfer pipelines — and reports the consortium-scale study that validates the approach across three physically distinct robot platforms.

9.1 Robot platforms and evaluation tasks

The consortium study replicates the same evaluation protocol across three robot platforms chosen for their physical diversity:

- **Robot A** — two 6-DoF ARX5 arms with parallel-jaw grippers mounted *upright* on an aluminium frame. Main egocentric camera: Aria glasses. Wrist cameras: Intel RealSense D405.
- **Robot B** — two ARX5 arms *side-mounted* on a custom 3D-printed shoulder structure that approximates the human workspace. Main egocentric camera: head-mounted Aria. Wrist cameras: Logitech webcams.
- **Robot C** — Unitree G1 humanoid with 7-DoF arms, each equipped with a 6-DoF dexterous hand. Main egocentric camera: ZED 2 stereo.

Evaluation runs on four of the six HANVERSE-A flagship tasks (Section 4.3): `object-in-container`, `cup-on-saucer`, `bag-grocery`, `fold-clothes`. Per platform and per task, 20 in-domain (ID) rollouts and 20 out-of-domain (OOD) rollouts are run; OOD swaps in unseen objects and novel environments. Results are reported as normalized scores aggregated across rollouts, with task-specific subtask metrics (grasp, placement, full completion). Robot demonstrations are collected at 150–300 demos per task with 4–8 objects per task.

9.2 Cross-embodiment learning architecture

To enable joint training across diverse embodiments, recent work adopts encoder-decoder architectures with shallow, modality-specific stems [9]. Image observations are processed by a visual backbone (e.g., ResNet-18 [28]), while proprioceptive inputs are encoded with an MLP before being tokenized into a shared space via learned query attention.

A shared vision stem processes egocentric RGB observations from both human and robot embodiments, while separate stems handle robot-specific wrist cameras and proprioception. The resulting tokens are concatenated and passed through a shared transformer encoder f_ϕ. Learnable tokens attend to the multi-modal inputs to extract task-relevant features and condition the action decoders. This architecture is illustrated in Fig. 13.

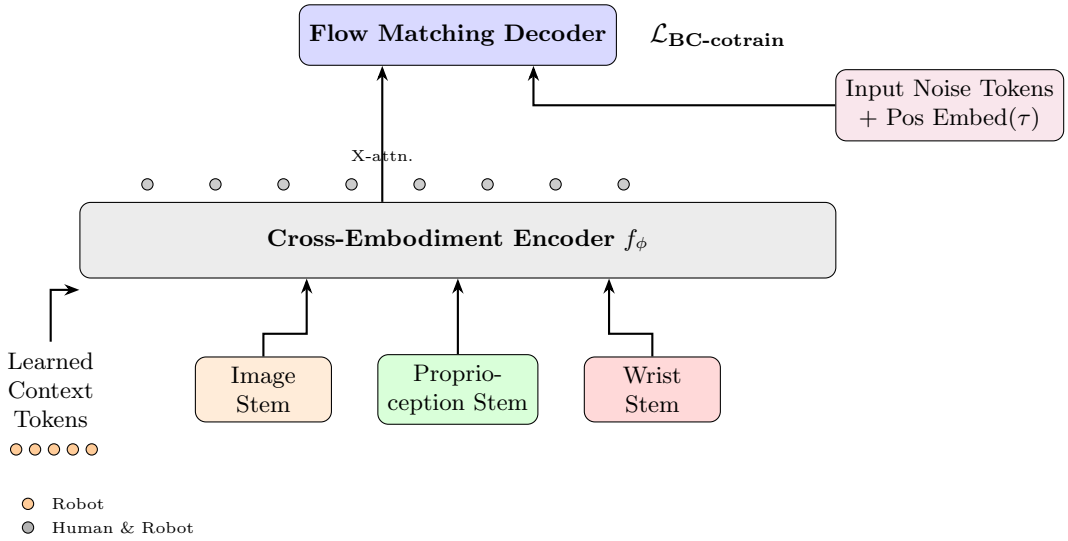


Figure 13: Model architecture. A transformer-based cross-embodiment policy backbone processes multi-modal inputs (egocentric images, proprioception, wrist cameras) through modality-specific stems. Learned context tokens distinguish embodiment types. A flow matching decoder produces action predictions, trained with the co-training loss across human and robot data.

9.3 Human and robot data alignment

Human egocentric hand tracking is typically performed in a moving camera frame. For joint policy learning, hand poses are projected into camera-centered stable reference frames. The raw trajectory of 3D hand poses $[p_t^H, p_{t+1}^H, \ldots, p_{t+k}^H]$ in the device frame T_t^{device} is transformed into a relative action representation:

$$a_{t:t+k}^H = \left[\left(T_t^{\text{device}} \right)^{-1} \cdot T_{t+i}^{\text{device}} \cdot p_{t+i}^H \right]_{i=1}^{k}. \tag{11}$$

To make normalization robust to outliers, quantile normalization maps the 1st and 99th percentiles of the feature distribution to the range $[-1, 1]$. For a feature tensor x, the normalized output $\hat{x}$ is:

$$\hat{x} = 2 \cdot \left(\frac{x - q_{0.01}}{q_{0.99} - q_{0.01}} \right) - 1. \tag{12}$$

9.4 Training objective

The encoder f_ϕ and action decoder π_θ are jointly optimized with a behavior cloning co-training loss computed on the aggregated human and robot dataset:

$$\mathcal{L}_{\text{BC-cotrain}}(\phi, \theta) = \mathbb{E}_{(o,a)\sim\mathcal{D}_H\cup\mathcal{D}_R} \left[\mathcal{L}_{\text{BC}} \big(\pi_\theta(f_\phi(o)),\, a \big) \right]. \tag{13}$$

In practice, per training step, for each embodiment $e \in \{\text{robot}, \text{human}\}$, the conditional flow matching (CFM) loss is computed on a mini-batch of human and robot samples:

$$\mathcal{L}_{\text{BC-cotrain}} = \mathcal{L}_{\text{CFM}}^{\text{robot}} + \mathcal{L}_{\text{CFM}}^{\text{human}}. \tag{14}$$

9.5 Key consortium findings

Three consistent findings emerge across the consortium study and inform HanVerse's design:

(1) Co-training improves transfer. Adding HanVerse-A data to the robot-only training set consistently improves both in-domain and out-of-domain performance across all three robot platforms and all four evaluation tasks. The OOD lift is the more important number — co-training yields up to a 30 percentage-point improvement in OOD success rate, well beyond noise. Figure 14 plots the ID and OOD success-rate panels. A single notable exception: Robot B's performance on `bag-groceries` decreased with co-training, traced to an embodiment-specific strategy divergence (Robot B uses one gripper to prop the bag open while the other inserts items, whereas human demonstrators and Robot A use two hands to open the bag). The divergence weakens cross-embodiment alignment during co-training for that specific platform-task pair.

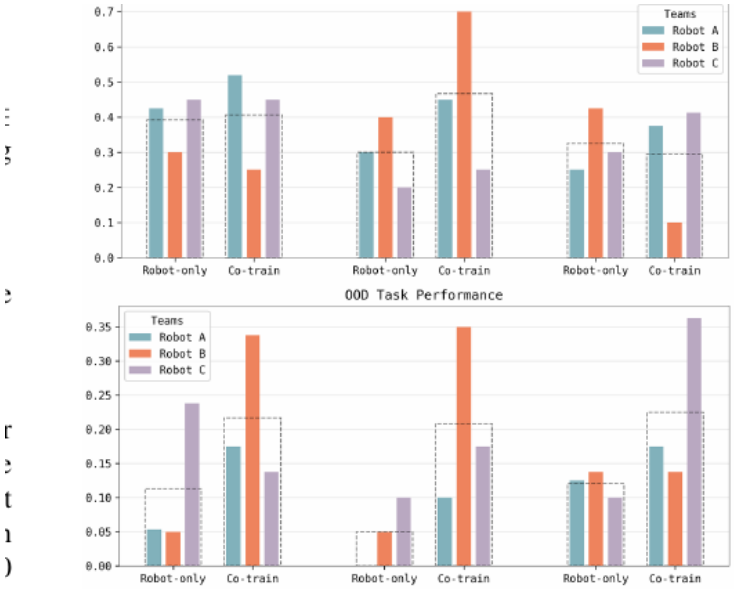


Figure 14: Co-training improves transfer. Joint training with HanVerse-A data consistently improves in-domain and out-of-domain task performance across three physically distinct robot platforms (Robot A: ARX5 upright; Robot B: ARX5 side-mounted; Robot C: Unitree G1 humanoid). Top: OOD task performance. Bottom: in-domain task performance.

(2) Domain-aligned data anchors scaling. Neither large volumes of diverse HanVerse-A data nor domain-aligned data alone drive significant gains. Positive scaling emerges only when a small amount of domain-aligned data is included alongside the diverse human corpus. Concretely: just 2 hours of domain-aligned human data unlocks transfer from 2 hours of diverse HanVerse-A data, and performance scales further as the diverse pool grows to 8 hours. The relationship is strongly non-linear — diverse data without alignment anchors yields minimal improvement. This finding is the empirical reason the Humanoid Network treats HanVerse-A (the reproducible academic layer) as strategically essential even though it is 20× smaller than HanVerse-I by hours.

(3) Scene diversity dominates for environment generalization; demonstrator diversity for embodiment generalization. Controlled-diversity ablations (Section 4.4) show that

under limited budgets, scaling the number of scenes is the single most reliable driver of generalization to unseen environments — Avg-MSE drops roughly 22% going from 1 scene to 16 scenes at a fixed total budget. Scaling the number of demonstrators instead gives more modest but robust gains, and its main contribution is robustness to unseen human morphologies. Once data quantity reaches a moderate level, increasing *density* within the same scene/demonstrator distribution yields diminishing returns — it is coverage, not volume, that matters.

Figure 15 shows controlled diversity experiments measuring the effect of scaling demonstrator count and scene count on downstream policy performance, holding total data budget constant. The results demonstrate that both forms of diversity contribute to generalization, with scene diversity playing a particularly strong role under limited data budgets.

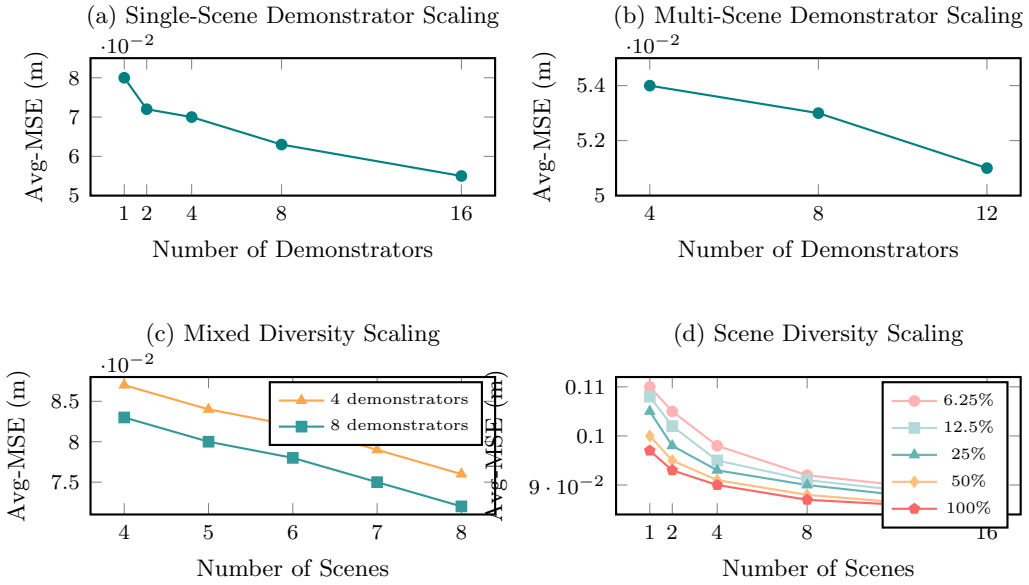


Figure 15: Controlled diversity results. (a) Scaling demonstrators in a single scene improves generalization to unseen ones. (b) Demonstrator scaling remains beneficial across multiple fixed scenes. (c) Jointly scaling scene and demonstrator diversity yields complementary improvements. (d) Increasing scene diversity improves generalization to unseen scenes across data budgets, with the strongest gains under limited data.

These findings directly inform HanVerse's design. The diversity-weighted quality scoring (Eq. (6)) prioritizes underrepresented scenes and demonstrator profiles. The task taxonomy (Section 4.7.1) ensures alignment between human demonstrations and target robot tasks. And the global contributor base naturally produces the scene and demonstrator diversity that controlled studies have shown to be most valuable for policy generalization.

The relationship between dataset scale n, domain alignment α, and downstream policy performance P can be approximated as:

$$P(n, \alpha) = P_0 + \beta \cdot n^{\gamma} \cdot f(\alpha), \tag{15}$$

where P_0 is the robot-only baseline, γ is the scaling exponent, and $f(\alpha)$ captures the quality-adjusted contribution of aligned versus unaligned data. The key insight from empirical work is that $f(\alpha)$ is strongly non-linear: a small amount of domain-aligned data unlocks scaling from diverse data, while diverse data alone without alignment anchors yields minimal gains.

10 $HAN, veHAN, and Future Protocol Eligibility

The current launch framework separates the canonical HAN token from Humanoid Network's protocol infrastructure. The canonical **$HAN** token contract is expected to be created on Base at TGE. Humanoid Network's own contracts and databases then use that official token address for locks, eligibility records, Motion Credits, burn or sink accounting, and future claim controls. This paper does not disclose final offering documents, public allocation language, or claim schedules. Final launch parameters will be published in definitive launch documents.

10.1 Canonical HAN and protocol contracts

The **$HAN** token created at TGE is the canonical asset. It is not the Humanoid Network protocol reward contract. Humanoid Network therefore needs separate launch infrastructure: an official HAN token controller, a veHAN lock vault, a proof and eligibility registry, a Motion Credit manager, burn or sink accounting, and a future reward vault whose claim and payout functions remain disabled until approved.

10.2 veHAN locks

veHAN is non-transferable participation power created when a user locks official **$HAN** in an approved lock vault. Longer locks can produce greater participation weight, but the v0 lock vault does not burn principal, does not allow early unlock, and does not include a reward payout function. veHAN may activate reward-eligible submission lanes, eligibility records, and governance-adjacent participation after those lanes are approved.

10.3 Motion Credits

Motion Credits are non-transferable capacity credits. They can increase the number of reward-eligible submissions a participant can make during a configured epoch, but they do not directly buy rewards. Motion Credit purchases may route HAN into burn, sink, contributor, compute, anti-abuse, infrastructure, or future staker pools according to configuration. Because the canonical HAN token may or may not expose a `burn()` function, burn accounting must support both direct burns and transfers to an immutable burn or sink address.

10.4 Future protocol reward eligibility

Eligibility records can combine veHAN power, lock duration, days since TGE, accepted motion hashes, Hydra scores, novelty or duplicate status, Motion Credits purchased, and HAN burned or sunk. These records are not claimable rewards. They are future protocol reward weight only. Claims remain disabled until reserve unlocks, app capacity, Hydra scoring, anti-abuse controls, final contracts, and approvals are complete.

10.5 $HAN token utility

At the Token Generation Event, **$HAN** becomes the coordination unit of the broader protocol economy. Primary utilities may include protocol access, veHAN participation, Motion Credit purchase, enterprise settlement, and future governance. None of those utilities should be read as a promise of APY, yield, revenue share, guaranteed rewards, guaranteed returns, buybacks, or price appreciation.

10.6 Issuance discipline

Token issuance is intentionally not pinned to a fixed emission schedule in this paper. The design considerations are well known from prior data-network and contribution-reward protocols [29, 30]: emissions should be responsive to genuine contribution quality and utilization (so they do not flatly inflate during spam-heavy periods) and bounded on the upside (so no single period can dilute prior contributors beyond governance-approved limits). The Humanoid Network treats the specific functional form of emission, the base rate, and any adjustment parameters as governance variables, set under the authority of token holders rather than by this paper.

10.7 Marketplace dynamics

Enterprise customers access Humanoid Network data through two primary commercial models. The first is per-dataset licensing, where customers purchase curated bundles filtered by task category, physics-validated quality threshold, diversity profile, and annotation depth. The second is subscription access, providing continuous streaming of new data as it enters the corpus. Enterprise payments may be made in fiat or supported protocol assets. Any route from enterprise revenue into protocol rewards requires separate reserve unlocks, contracts, controls, and approvals.

11 App Capacity and Access Controls

Outside the token framework, the Humanoid Network may control app capacity through account approvals, allowlists, queue controls, task pools, rate limits, Motion Credits, and enterprise access permissions. These controls are product and infrastructure controls, not token reward promises.

11.1 What capacity controls may provide

Capacity controls may provide access to features that are expensive to operate or scarce by design:

- Queue access or rate-limited motion generation during high-demand periods.
- Early access to new robot target classes as they are added to the HYDRA validation pipeline.
- A private dataset viewer for deeper inspection of the corpus than the public explorer exposes.
- Scheduled teleoperation sessions on partner-owned hardware, subject to availability.
- Partner or enterprise workflow permissions.

No **$HAN** is issued in exchange for access permissions, and access controls do not create claimable rewards.

11.2 What capacity controls do *not* do

Capacity controls do *not* grant a preferential reward rate, do *not* guarantee future protocol rewards, and do *not* bypass attestation, quality scoring, duplicate detection, or physics validation. A participant who submits the same motion should receive the same HYDRA score, proof result, and quality treatment regardless of access path.

This separation is deliberate. Capacity is a product and infrastructure decision. A preferential path to future token rewards would change the legal and trust posture of the protocol; see Section B for the relevant reasoning.

12 Network Metrics and Forward Evaluation

Section 9 reports the consortium-scale validation of the core scientific claim: human egocentric data co-trained with robot data improves generalization. This section describes the operational metrics the network surfaces in production, plus the forward experiments planned as the dataset and participation base grow.

12.1 Network-health metrics

The protocol tracks: contributor count and geographic distribution, episodes accepted per task category, average quality score by category and region, task completion rate and time-to-completion, reviewer accuracy and consensus rates, HYDRA-Track MPJPE and foot-skate distributions over recent submissions, generation infrastructure health, storage capacity, veHAN participation, Motion Credit usage, and burn or sink accounting once those systems are live. A subset of these is surfaced publicly in the app; sensitive operational metrics remain internal.

12.2 Forward experiments

The following experiments are planned as new data and new target embodiments onboard:

Embodiment scaling. As new humanoid target classes (beyond Unitree G1) enter the HYDRA-Track validation pipeline, we will re-run the consortium transfer study on each and report cross-embodiment transfer curves. The hypothesis, consistent with Section 9, is that the domain-aligned-anchor phenomenon holds per-embodiment — a small amount of embodiment-aligned data will be required to unlock the benefit of large cross-embodiment pools.

Diversity-targeted eligibility. The quality score $Q(\mathcal{E})$ (Eq. (6)) includes a diversity term $\Delta(\mathcal{E})$ intended to direct eligibility weight toward under-represented scene and demonstrator profiles. We will measure whether diversity-weighted task selection produces measurably better generalization in the downstream policies than flat task weights at matched data volumes.

Long-horizon refinement. Beyond co-training, we plan pre-train / fine-tune paradigms that use HANVERSE-I at pre-training scale and the matched aligned subset for fine-tuning, and will report whether this separates the diversity and alignment effects in a more actionable way than joint co-training.

Incentive effectiveness. Compare data quality distributions under protocol-weighted contribution versus flat-rate compensation, testing whether quality-weighted eligibility produces measurably better training data.

13 Roadmap

This roadmap is an indicative planning view of intended milestones and is not a representation that any listed capability is complete or available today.

Table 12: Indicative roadmap organized into four phases (non-binding planning view).

Phase	Components	Scope
Phase 1: Core Product	App, backend API, data pipeline, public motion browser	HANCAPTURE and app-based motion creation; backend API with auth and episode ingestion; HANDB processing pipeline; public Create, Motions, and Dataset surfaces.
Phase 2: Token Launch Prep	Smart contracts, veHAN, quality pipeline, evaluation	Base Sepolia rehearsal and audit; official HAN controller; veHAN lock vault; eligibility registry; Motion Credits; full HANSCORE quality pipeline; HYDRA physics-validation coverage across target embodiments.
Phase 3: TGE	Canonical token deployment, app rehearsal, enterprise API	Canonical HAN token created on Base at TGE; official contract verification; Buy HAN route; lock and eligibility rehearsal if approved; reward claims disabled; enterprise dataset API; first enterprise engagements.
Phase 4: Expansion	Capacity controls, additional robot embodiments, partner rigs	Expansion of HYDRA validation to additional humanoid target classes; onboarding of new partner data rigs; enterprise workflow tooling; future claim-controller activation only after reserves, capacity, anti-abuse, contracts, and approvals are complete.

14 Related Work

Datasets of human activities. Large-scale human activity datasets such as Something-Something V2 [15], Ego4D [4], and Epic-Kitchens [5] capture rich human behavior across diverse environments. However, they are not designed for robot learning: they often include tasks beyond current robot capabilities, lack manipulation-relevant annotations such as precise hand poses or object interactions, and contain unstructured activities that are difficult to translate into executable robot demonstrations. HANVERSE emphasizes bounded diversity, focusing on tasks that are feasible for typical bimanual mobile manipulators while preserving natural variation across environments, objects, and demonstrators.

Robot learning from human data. Human data presents opportunities for robot learning through both abundant unlabeled online videos and curated, labeled demonstrations [10, 11]. Labeled human demonstrations can be co-trained with robot data as distinct embodiments for policy learning, post-training, and world modeling [9, 31]. These works show that co-training enhances robustness and scene understanding. HANVERSE builds on these findings by providing the large-scale, diverse human data that these methods require.

Scaling robot learning with massive data. Public efforts such as Open X-Embodiment [1], DROID [32], and Rh20t [33] demonstrate that training on diverse, multi-embodiment data improves generalization across tasks and environments. However, achieving generally capable robots remains

fundamentally constrained by data scalability. HanVerse examines how human egocentric data can support robot learning at scale by treating it as a first-class data source alongside robot data.

Crowdsourced robot data. RoboTurk [34] pioneered crowdsourced teleoperation through a web interface, and RoboNet [35] demonstrated multi-robot learning across institutions. HanVerse extends the crowdsourcing paradigm from robot teleoperation to human demonstration capture, removing the requirement for physical robot access and using protocol eligibility instead of per-task payments through institutional channels.

Token-incentivized data networks. The intersection of cryptoeconomics and data collection is explored in recent work on tokenized incentives for federated learning [29] and data marketplace pricing [30, 36]. Several recent projects in the Web3 space have applied token-incentivized data networks to coordinate distributed contribution for AI training, demonstrating that adaptive emission mechanics and cryptographic sybil resistance can sustain quality in decentralized data marketplaces. HanVerse builds on these foundations with the specific requirements of egocentric human data collection: phone-based capture, cross-embodiment transfer, and diversity-weighted quality scoring.

15 Conclusion

We have introduced HanVerse, a protocol for scalable human data-driven robot learning that combines egocentric capture, cloud-based data management, automated quality scoring, and token-incentivized global contribution. The core thesis is that the science of human-to-robot transfer has matured to the point where the binding constraint is no longer algorithmic but operational: how to collect diverse human demonstration data across jurisdictions, demographics, and environments without the overhead of per-country employment infrastructure.

HanVerse addresses this by replacing institutional hiring with proof-gated protocol participation, enabling a contributor base that is limited only by smartphone penetration and supported wallet access rather than university affiliation. The diversity-weighted quality scoring system ensures that the data is not just large but representative, giving additional eligibility weight to contributions from underrepresented regions and demographics that existing centralized approaches structurally cannot reach.

Beyond the data infrastructure, HanVerse creates a transition economy for workers displaced by physical AI. The people who currently perform the manipulation, logistics, and service tasks that robots will automate are the most qualified to demonstrate those tasks for training data. By giving them a way to turn embodied expertise into proofed protocol participation, HanVerse aligns the economics of automation with the human side of the transition.

A Supplementary Material

A.1 Scaling experiment data budgets

Tables 13 to 15 detail the data budget allocation for the controlled diversity experiments presented in Fig. 15.

Table 13: Single-scene demonstrator scaling (fixed 2-hour budget). As the number of training demonstrators increases, the per-demonstrator data duration decreases proportionally to maintain the total budget.

# Train Demonstrators	Mins / DS Pair
1	120.0
2	60.0
4	30.0
8	15.0
16	7.5

Table 14: Multi-scene demonstrator scaling (fixed 8-hour budget). Training data are collected across 8 fixed scenes, with per-demonstrator duration decreasing as demonstrator count increases.

# Train Demonstrators	Mins / DS Pair
4	15.0
8	7.5
12	3.75

Table 15: Scene diversity scaling data composition. Total training budget (minutes) for different scene counts and data usage fractions (relative to 60 min/scene at 100%). All models are evaluated on unseen demonstrators at unseen scenes.

# Scenes	6.25%	12.5%	25%	50%	100%
1	3.75	7.5	15	30	60
2	7.5	15	30	60	120
4	15	30	60	120	240
8	30	60	120	240	480
16	60	120	240	480	960

All values are minutes of recording time for training data.

B Legal Characteristics of the $HAN Token

This appendix states the position that the Humanoid Network takes on the legal character of **$HAN** and related protocol mechanics. It is not legal advice, and it is not the product of definitive offering documents, which would include full disclosure of material information and risk factors and would supersede any statements here. It is a plain-English description of design intent.

B.1 Utility, not equity

$HAN is intended to be a utility token used to coordinate the Humanoid Network protocol: accessing approved protocol functions, locking into veHAN participation power, purchasing Motion Credits when enabled, supporting enterprise settlement paths, and participating in future governance if approved. **$HAN** is not equity in any entity. Holders have no ownership interest in Humanoid Network Inc. or any affiliated foundation, no claim on residual assets in a winding-up, and no guaranteed right to profits, revenue, yield, rewards, buybacks, or price appreciation. **$HAN** is not debt. No entity is obligated to repay holders, pay interest, or redeem **$HAN** at any fixed price.

B.2 Eligibility records are not claimable rewards

Eligibility records, proof attestations, veHAN power, and Motion Credit capacity do not by themselves create claimable rewards. They cannot guarantee any payment in fiat or cryptocurrency. They are records of participation, proof quality, capacity use, and future protocol weight, subject to app capacity, reserve unlocks, scoring rules, anti-abuse controls, final contracts, governance, and legal approvals.

B.3 Capacity access is not a token sale

The capacity controls described in Section 11 provide access to product features such as queues, private dataset views, partner workflows, and teleoperation sessions. They do not sell **$HAN** or any right to future **$HAN**. They do not guarantee future protocol rewards, and they do not bypass HYDRA scoring, duplicate detection, proof validation, or anti-abuse controls. Any fiat or stablecoin fee for product access would have the same character as a software-as-a-service or enterprise-access fee.

This separation is deliberate. A payment that visibly tied access fee to future token rewards would change the legal character of the fee under several jurisdictions' securities and commodities regimes. The Humanoid Network has no intention of operating such a structure.

B.4 Jurisdictional posture

Humanoid Network Inc. is incorporated in Delaware. Under the current launch framework, the canonical **$HAN** token contract is expected to be created on Base at TGE. No offering of **$HAN** is authorized or solicited by this White Paper. Any offering or listing-related document, if made, will be made only pursuant to definitive documents and applicable law in each jurisdiction where it is made.

B.5 No allocation, schedule, or guarantees

Consistent with the disclaimer on the title page, this paper is not the definitive source for token allocation percentages, vesting schedules, total supply, emission rates, or other launch parameters. Those parameters remain subject to definitive launch documents. Any figure quoted in third-party material before definitive documents should be treated as non-authoritative.

Nothing in this paper guarantees the success of the Humanoid Network protocol, the utility of **$HAN**, the value of veHAN participation, the timing of any Token Generation Event, the availability of any future claim function, or the occurrence of any milestone in the indicative roadmap. Forward-looking statements reflect current intent based on current information; they may prove incorrect. Prospective Contributors, veHAN participants, Enterprise buyers, and holders should perform their own diligence and consult their own professional advisors before engaging with the protocol.

References

[1] Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models, 2023.

[2] Anthony Brohan et al. RT-2: Vision-language-action models transfer web knowledge to robotic control, 2023.

[3] Yuhan Hu et al. Scaling data collection for robot learning with human-in-the-loop, 2024.

[4] Kristen Grauman et al. Ego4D: Around the world in 3,000 hours of egocentric video, 2022.

[5] Dima Damen et al. Scaling egocentric vision: The EPIC-KITCHENS dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[6] Jakob Engel et al. Project Aria: A new tool for egocentric multi-modal AI research, 2023.

[7] Mohtasim Hoque et al. EgoDex: Learning dexterous manipulation from large-scale egocentric hand-object interactions, 2025.

[8] Prithviraj Banerjee et al. HOT3D: Hand and object tracking in 3D from egocentric multi-view videos, 2024.

[9] Cheng Chi et al. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots, 2024.

[10] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild, 2022.

[11] Shengjie Cai et al. Learning manipulation from egocentric human videos without robot demonstrations, 2025.

[12] Daron Acemoglu and David Autor. Skills, tasks and technologies: Implications for employment and earnings. *Handbook of Labor Economics*, 4:1043–1171, 2011.

[13] Daron Acemoglu and Pascual Restrepo. Automation and new tasks: How technology displaces and reinstates labor. *Journal of Economic Perspectives*, 33(2):3–30, 2019.

[14] Daron Acemoglu et al. Tasks, automation, and the rise in U.S. wage inequality, 2024.

[15] Raghav Goyal et al. The "something something" video database for learning and evaluating visual common sense, 2017.

[16] Kevin Black et al. π_0: A vision-language-action flow model for general robot control, 2024.

[17] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[18] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[19] Unitree Robotics. Unitree G1 humanoid robot — product specification. https://www.unitree.com/g1, 2024. Manufacturer product page.

[20] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. DeepMimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 37(4), 2018.

[21] Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. AMP: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 40(4), 2021.

[22] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 34(6), 2015.

[23] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations (ICLR)*, 2024.

[24] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[25] NVIDIA et al. Isaac Lab: An open-source unified framework for robot learning. https://isaac-sim.github.io/IsaacLab/, 2025. GPU-accelerated robotics simulation framework.

[26] 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)*, 2012.

[27] Johan Björck et al. GR00T N1.5: An open foundation model for generalist humanoid robots. https://developer.nvidia.com/isaac/gr00t, 2025. Vision-language-action foundation model for humanoid robots.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[29] Shashi Raj Pandey et al. FedToken: Tokenized incentives for data contribution in federated learning, 2022.

[30] Zongyang Wang et al. Cryptoeconomics and tokenomics as economics, 2024.

[31] Tairan He et al. Learning cross-embodiment manipulation policies from egocentric human videos, 2025.

[32] Alexander Khazatsky et al. DROID: A large-scale in-the-wild robot manipulation dataset, 2024.

[33] Hao-Shu Fang et al. RH20T: A comprehensive robotic dataset for learning diverse skills in one-shot, 2023.

[34] Ajay Mandlekar et al. RoboTurk: A crowdsourcing platform for robotic manipulation, 2018.

[35] Sudeep Dasari et al. RoboNet: Large-scale multi-robot learning, 2019.

[36] Mengxiao Zhang, Fernando Beltran, and Jiamou Liu. A survey of data pricing for data marketplaces, 2023.