Generative Video for Humanoid Control

Matei Gardea[*]
University of California, Berkeley

Koushil Sreenath
University of California, Berkeley

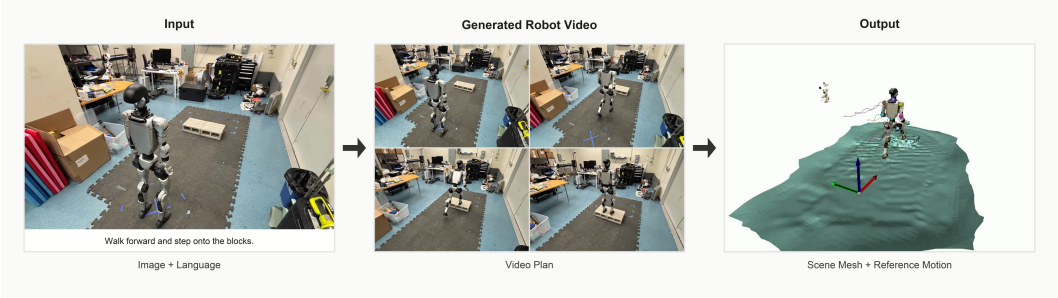

Figure 1: GVHC takes an initial image and a language instruction as input, generates a video plan, and converts the generated video into scene geometry and a reference motion.

Abstract: A central challenge in robot learning is converting high-level intent into physically executable behavior. Generative video models (GVMs), such as Veo 3.1, have started to show signs of understanding basic physical behavior in visual prediction tasks. This suggests that videos synthesized from language and images may encode useful cues about how objects, agents, and environments evolve under plausible real-world dynamics. This observation is particularly relevant for robotics: a generated task video can reveal the spatial layout, object interactions, and agent motion implied by an instruction, reducing language-conditioned control to the recovery of control-relevant structure from video. In this work, we study a simple route for grounding synthesized videos of a robot acting in a static scene into explicit geometric and kinematic structures for control. The input is a natural-language instruction and a single exocentric RGB image of a Unitree G1 in its starting configuration. The output is a metric scene representation and a parameterized robot reference trajectory, which together define the simulation scene and target behavior for a tracking policy. The current prototype implements the scene-reconstruction side of this pipeline, while robot pose estimation, policy training, and physical deployment are in progress.

Keywords: Whole-body control, visual imitation, video diffusion models, scene geometry reconstruction, motion tracking

1 Introduction

A central challenge in robot learning is converting high-level intent into physically executable behavior. Natural language is expressive but underspecified: an instruction such as "walk forward and step onto the blocks" does not directly specify a footstep sequence, body posture, contact timing, camera-relative geometry, or the local obstacle shape. In contrast, a video of the task contains many of these missing details. It shows where the robot moves, which surfaces are relevant, which paths through the scene are plausible, and how the body evolves over time.

Recent generative video models make it possible to synthesize such videos from language and images. Models such as Veo 3.1 can generate short image-conditioned videos [1], and recent studies of

[*]Correspondence: matei.gardea@berkeley.edu

video models as zero-shot visual reasoners suggest that these models encode nontrivial knowledge about segmentation, object affordances, and physical evolution [2]. For robotics, this raises a useful question: if a generated task video is not itself an executable policy, can it still be converted into control-relevant structure?

We explore this question in the setting of humanoid locomotion and navigation. The input is a natural-language instruction and a single exocentric RGB image of a Unitree G1 in the target static scene. A generative video model produces a short video depicting the G1 carrying out the instruction. GVHC (Generative Video for Humanoid Control) converts the generated video into a metric static scene representation and sets up the planned recovery of a parameterized humanoid reference trajectory. Together, these outputs would specify the simulation scene and target behavior for a tracking policy.

This framing is intentionally narrower than general robot video understanding. We assume the scene is static except for the robot. We assume the generated video depicts the target robot embodiment rather than a human demonstrator. We focus on locomotion and navigation rather than manipulation or dynamic object interaction. These restrictions let us isolate a concrete technical problem: recovering a static scene representation suitable for physics simulation, and eventually a trackable robot motion, from a single generated monocular video.

The current prototype implements the scene side of the pipeline. It uses SAM 3 to segment the robot [3], MoGe-2 and GeometryCrafter to estimate per-frame point maps [4, 5], dense optical flow and robust 3D alignment to register static geometry across frames, manually selected scene keypoints to align the reconstruction to a metric gravity frame, and Poisson surface reconstruction to export a mesh intended for simulation [6, 7]. The reference trajectory stage is being developed: the intended module isolates the masked robot point cloud in each generated frame and estimates the G1 root pose and joint configuration. Policy training and physical deployment are also in progress.

This paper makes three contributions. First, it formulates generated robot video as an intermediate representation for language-conditioned humanoid control. Second, it describes a modular pipeline for turning the generated monocular video into explicit scene geometry and a planned robot reference-trajectory recovery stage. Third, it identifies the practical failure modes that arise when control depends on video-model consistency, monocular geometry, segmentation, and articulated pose recovery.

2 Related Work

Video generation as a robot world model. A growing body of work treats video generation as a substrate for robot planning and data generation. UniPi and related video-planning methods generate visual plans and then recover actions through inverse dynamics or low-level controllers [8, 9]. DreamGen uses video world models to synthesize robot videos and recovers pseudo-actions with latent action or inverse-dynamics models, showing that generated videos can support behavior and environment generalization [10]. These systems motivate our use of generated video as a source of task structure, but our output is different: rather than recovering actions directly from video, we recover explicit 3D scene geometry and formulate the planned humanoid reference-trajectory recovery problem.

Humanoid imitation from video. VideoMimic, ASAP, HumanPlus, and related systems show that single videos or human motion data can be converted into humanoid skills through reconstruction, retargeting, and physics-based tracking [11, 12, 13]. VideoMimic is especially relevant because it jointly reconstructs humans and environments from monocular videos and trains contextual humanoid control policies. Our problem differs in the source of the demonstration: the video is generated from language and a single image rather than captured from a real human. It also differs in embodiment: the generated video is intended to show the G1 itself, avoiding human-to-robot retargeting but requiring robot-specific pose estimation from generated pixels.

Geometry, segmentation, and pose estimation. MoGe-2 predicts metric 3D point maps from single images, while GeometryCrafter estimates temporally coherent point map sequences from open-world videos using diffusion priors [4, 5]. Dynamic reconstruction and SLAM systems such as MonST3R and DUSt3R-family models address related problems of camera registration and 3D structure recovery in the presence of motion [14, 15]. Promptable segmentation systems such as SAM 3 provide text-conditioned masks that let us exclude the robot from static reconstruction [3]. Robot pose estimation methods such as DREAM and RoboPose use rendered data and known robot models to estimate articulated robot state from images [16, 17]. Our planned pose-estimation module lies at the intersection of these areas: it must recover the root pose, joint configuration, and camera parameters of a G1-like articulated robot from generated RGB and masked point-cloud observations.

3 Problem Formulation

We consider a static scene containing a Unitree G1 humanoid in an initial configuration. The input is an RGB image I_0 from an exocentric camera and a natural-language instruction ℓ describing a locomotion or navigation behavior. The goal is to construct a simulation problem in which a G1 policy can learn to execute the behavior implied by ℓ in the geometry of the scene.

GVHC uses a generative video model G to produce a short video

$$V = \{I_t\}_{t=0}^{T-1}, \qquad I_t \in \mathbb{R}^{H\times W\times 3}, \tag{1}$$

conditioned on I_0 and ℓ. The generated video is not assumed to be dynamically correct in the strict sense. Instead, we treat it as a noisy visual plan that may contain useful spatial and kinematic cues. The recovery problem is to estimate (i) a static scene mesh M in a metric, gravity-aligned coordinate frame and (ii) a time-indexed G1 reference trajectory $q_{\mathrm{ref}}(t)$, including root pose and joint configuration.

The intended control stage imports M into simulation and trains a tracking policy $\pi(a_t \mid o_t, q_{\mathrm{ref}})$ to follow the recovered reference while respecting contacts and robot dynamics. This paper focuses on the recovery pipeline. In the current implementation, the scene mesh M is produced by the system, while q_{ref}, policy training, and real-world deployment are ongoing.

We make four assumptions. First, the scene is static; only the robot moves. Second, the generated video depicts the target robot embodiment, the Unitree G1. Third, the relevant behavior is locomotion or navigation, so the main environmental contacts are feet and terrain-like surfaces. Fourth, the generated video is short, currently on the order of six seconds, which reflects the operating regime used in our image-to-video experiments.

4 Method

The method decomposes generated-video grounding into six implemented stages: video generation, robot segmentation, monocular geometry prediction, static scene registration, metric gravity alignment, and mesh reconstruction. The final planned stage, robot pose estimation, is intended to convert the moving robot point cloud into a reference trajectory. We denote pixel coordinates by $p = (u, v)$, camera-frame point maps by $X_t(p) \in \mathbb{R}^3$, and registered world-frame point maps by $Y_t(p) \in \mathbb{R}^3$. Invalid point-map values are represented as missing observations.

4.1 Video Generation and Segmentation

We condition Veo 3.1 on the input instruction and the initial exocentric image. The implementation requests a six-second, 16:9 video at 720p resolution and 24 FPS. This stage produces only pixels; it does not expose camera motion, depth, object state, contacts, or robot joint angles.

We segment the robot in the generated video using SAM 3 with a text prompt for the humanoid robot. The output may contain one or more object masks per frame. The implementation collapses

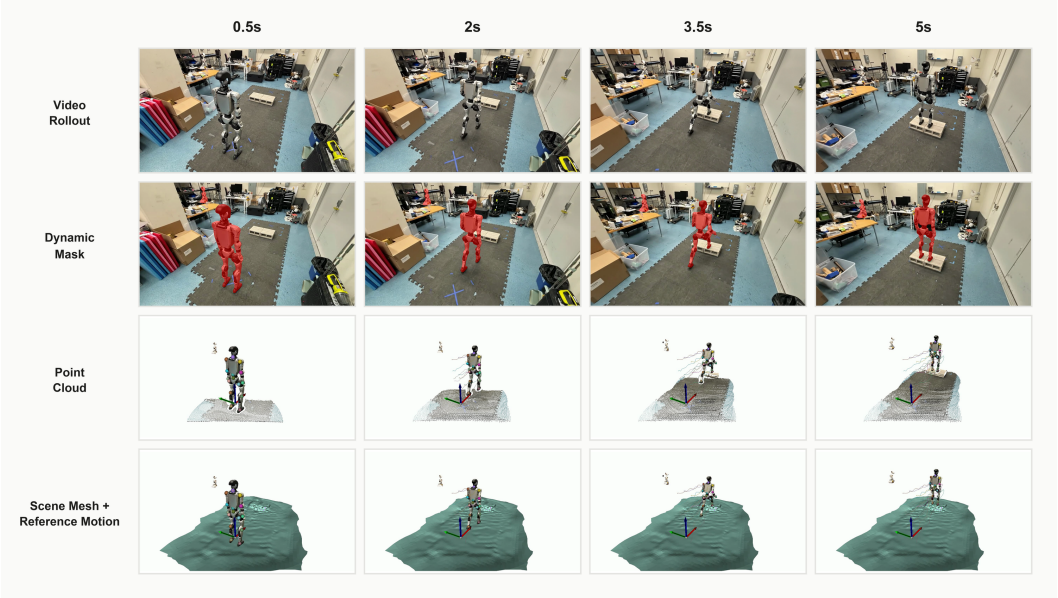

Figure 2: Qualitative outputs over the generated sequence. The video is segmented, lifted into a point-cloud representation, and shown with the reconstructed scene mesh and planned reference motion at matched timestamps.

them into a single dynamic mask

$$D_t(p) = \bigvee_{k=1}^{K} D_{t,k}(p) \in \{0, 1\}. \tag{2}$$

For scene reconstruction, D_t defines dynamic pixels that should be excluded from static registration and mesh aggregation. With a morphological dilation operator $\delta_r(\cdot)$, the static mask is

$$S_t(p) = \neg\delta_r(D_t)(p). \tag{3}$$

This conservative dilation removes uncertain boundary pixels around the robot, reducing the chance that moving limbs contaminate the static reconstruction.

4.2 Per-Frame Geometry

For each generated frame, we estimate a camera-frame point map. The pipeline first runs MoGe-2 to obtain per-frame point maps, metric scale estimates, and validity masks. The implementation converts MoGe's predicted geometry into camera-frame points by solving for a focal/shift correction and projecting depths through normalized intrinsics. It then uses the MoGe point maps to construct GeometryCrafter priors: normalized disparity, an xy/z point-map representation, an intrinsic map, and a validity mask. GeometryCrafter denoises these priors over the video and outputs temporally coherent depth, validity, and intrinsic predictions.

The result is a sequence $X_t(p)$ of 3D points in each frame's camera coordinate system,

$$X_t(p) = (x_t(p), y_t(p), z_t(p)). \tag{4}$$

At this point, each frame has geometry, but the frames are not yet registered into a common world coordinate frame.

4.3 Static Scene Registration

We estimate relative camera motion using only static scene pixels. For temporal gaps $g \in \{1, 2, 4, 8, 16, 32, 64\}$, the implementation considers frame pairs (a, b) where $b = a+g$. It computes

dense optical flow $F_{a\to b}$ and $F_{b\to a}$ using OpenCV DIS optical flow. A pixel p in frame a proposes a match

$$p' = p + F_{a\to b}(p). \tag{5}$$

The match is kept only if it lies in bounds, both points are static and valid, and the forward-backward cycle error is small:

$$\|p + F_{a\to b}(p) + F_{b\to a}(p') - p\|_2 < \tau_{\text{cyc}}. \tag{6}$$

The implementation also filters candidate pixels by depth percentile, keeping nearer valid scene points before sampling. The current configuration keeps the nearest 50 percent of valid depths, samples up to 3000 correspondences per edge, applies a 0.5-pixel cycle-consistency threshold, runs 300 RANSAC iterations, and requires at least 50 inliers. Each surviving correspondence gives a pair of 3D points $(X_a(p), X_b(p'))$, one in each camera frame.

For each pair (a, b), the relative pose is estimated with a RANSAC-wrapped Kabsch alignment. Given sampled 3D correspondences $\{(x_i, x_i')\}_{i=1}^{N}$, the pose solves

$$R_{ab}, t_{ab} = \arg\min_{R\in SO(3), t\in\mathbb{R}^3} \sum_i \rho_i \left\|Rx_i + t - x_i'\right\|_2^2, \tag{7}$$

where ρ_i denotes the robust inlier weight. RANSAC selects inliers under a 0.02-meter residual threshold in the current metric frame and the final refinement uses iterative reweighting. Edges with fewer than the configured minimum number of inliers are discarded.

The remaining multi-gap edges form a pose graph over the video. Let $T_t \in SE(3)$ denote the world-to-camera transform for frame t, and let $\hat{T}_{ab}$ be the measured relative transform from frame a to frame b. The implementation initializes T_t by breadth-first traversal over the edge graph and then optimizes a robust least-squares objective of the form

$$\min_{\{T_t\}_{t=1}^{T-1}} \sum_{(a,b)} w_{ab} \left\|\log\left(T_b T_a^{-1} \hat{T}_{ab}^{-1}\right)\right\|_2^2, \tag{8}$$

with T_0 fixed to identity. The edge weights increase with inlier count and decrease with median residual, matching the code's $\sqrt{n_{\text{inlier}}}/r_{\text{median}}$ weighting. After optimization, camera positions and rotation vectors are smoothed with a Savitzky-Golay filter.

Finally, each camera-frame point is transformed into the shared world frame. Since $T_t = [R_t \mid t_t]$ is stored as a world-to-camera transform, the world point is

$$Y_t(p) = R_t^\top (X_t(p) - t_t). \tag{9}$$

4.4 Gravity Alignment and Mesh Reconstruction

Control requires a coordinate system with a physically meaningful vertical axis and metric scale. The current prototype uses manually selected scene keypoints in the first frame to define this alignment. The selected pixels are lifted through Y_0, producing 3D points p_1, p_2, p_3. The code uses p_1 and p_2 to define a horizontal reference direction and p_3 to define up from their midpoint:

$$o = \frac{1}{2}(p_1 + p_2), \qquad \hat{z} = \frac{p_3 - o}{\|p_3 - o\|_2}. \tag{10}$$

It then removes the vertical component of $p_1 - o$ to form a horizontal axis,

$$\tilde{y} = (p_1 - o) - ((p_1 - o)^\top \hat{z})\hat{z}, \qquad \hat{y} = \frac{\tilde{y}}{\|\tilde{y}\|_2}, \tag{11}$$

and defines $\hat{x} = \hat{y} \times \hat{z}$. The rigid alignment matrix is

$$E = \begin{bmatrix} \hat{x}^\top \\ \hat{y}^\top \\ \hat{z}^\top \end{bmatrix}, \qquad b = -Eo. \tag{12}$$

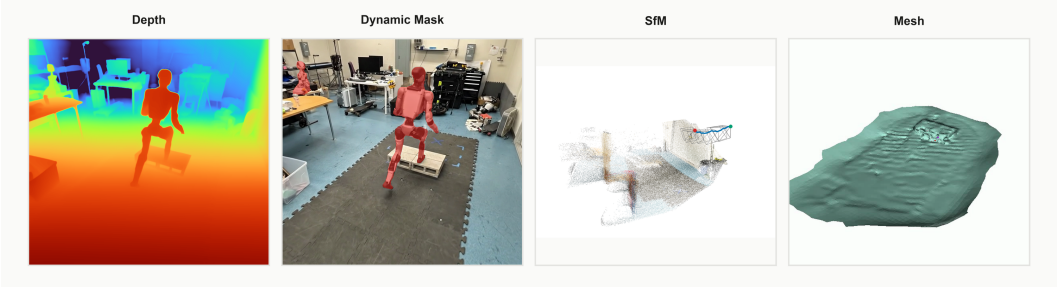

Figure 3: Static scene reconstruction. The generated video is lifted into depth, dynamic robot pixels are segmented, static geometry is registered with flow-based 3D alignment, and the fused local mesh is recovered for simulation.

A known distance d fixes scale using $s = d/\|p_3 - o\|_2$. With a configured offset c, each registered world point is mapped into the gravity-aligned metric frame as

$$Z_t(p) = s(EY_t(p) + b) + c. \tag{13}$$

The aligned static points are aggregated across frames to create a local scene mesh. For each frame, the implementation first defines valid static pixels

$$A_t = \{p : Z_t(p) \text{ is finite and } S_t(p) = 1\}. \tag{14}$$

It also computes the median robot position in the horizontal plane from valid dynamic pixels,

$$m_t = \text{median}_{p:D_t(p)=1}\ Z_t(p)_{xy}. \tag{15}$$

The kept static points are those within a configured horizontal radius r of the robot:

$$\mathcal{P} = \bigcup_t \{Z_t(p) : p \in A_t,\ \|Z_t(p)_{xy} - m_t\|_2 < r\}. \tag{16}$$

The current configuration uses a one-meter horizontal radius around the robot, 0.02-meter voxel downsampling, statistical outlier removal with 20 neighbors and standard-deviation ratio 2.0, normal estimation with 0.1-meter search radius and 30 neighbors, and Poisson reconstruction at depth 7. Low-density vertices are removed by a 5 percent density quantile threshold, unreferenced vertices are removed, only the largest connected triangle component is retained, and 10 iterations of Taubin smoothing are applied. The final output is an STL mesh.

4.5 Robot Pose Estimation and Reference Recovery

This stage is in progress. The intended input is the masked robot point cloud for each generated frame,

$$\mathcal{R}_t = \{Z_t(p) : D_t(p) = 1,\ Z_t(p) \text{ is finite}\}, \tag{17}$$

together with the RGB crop and camera geometry recovered by the earlier stages. The target output is a G1 root pose $T_t^{\text{root}} \in SE(3)$ and joint configuration $q_t \in \mathbb{R}^{n_q}$ for each frame. These estimates will be temporally assembled into a parameterized reference trajectory

$$q_{\text{ref}}(t) = (T_t^{\text{root}}, q_t). \tag{18}$$

The design goal is not to reconstruct a human body and retarget it. Because the generated video depicts the G1 directly, the pose estimator can use the G1 kinematic model and joint limits. This avoids one source of ambiguity but introduces another: current vision models and pose-estimation datasets are much stronger for humans than for humanoid robots. We are therefore treating robot pose estimation as a dedicated module rather than assuming that off-the-shelf human pose recovery will transfer.

5 Current Prototype and Qualitative Validation

The current repository contains an implemented prototype for generated-video scene reconstruction. The saved example data consists of a six-second, 144-frame, 1280 by 720 video processed through segmentation, MoGe-2, GeometryCrafter, static registration, gravity alignment, and mesh reconstruction. The pipeline saves rendered videos for each intermediate stage and exports a static mesh.

The qualitative validation at this stage is procedural rather than benchmark-based. The intermediate renderings are used to inspect whether SAM 3 tracks the robot mask through the video, whether MoGe-2 and GeometryCrafter produce coherent local point maps, whether the static registration stabilizes scene geometry across frames, whether the gravity-aligned frame is plausible, and whether the final Poisson mesh captures the local terrain around the robot. This is the appropriate level of evidence for the current prototype: benchmarks for generated-video-to-humanoid-control do not yet exist, and the pose and policy stages are being developed.

The prototype also exposes several informative failure modes. If the generated video changes the scene layout over time, the static registration stage receives inconsistent geometry. If the robot mask misses moving limbs, those limbs can appear as ghost geometry in the mesh. If the generated camera motion is large or texture-poor regions dominate, optical flow correspondences may be insufficient for reliable pose-graph connectivity. If the monocular point maps have scale drift, the manual gravity alignment can correct only global scale, not local geometric inconsistency. These issues show where generative video quality begins to constrain physical robot control.

6 Discussion, Limitations, and Future Work

The main question raised by this project is whether generated video becomes useful for robot control once it is made explicit. The simplicity of the representation is the point: instead of asking a video model to output actions, GVHC asks it for an imagined task video, then recovers the quantities that a locomotion policy can use: geometry, contacts, root motion, and joint targets. This separates semantic task imagination from physically grounded control while preserving a direct path back to simulation.

This explicit decomposition has several advantages. It allows the system to use strong off-the-shelf vision models for segmentation and geometry, while keeping the control problem in simulation rather than pixel space. It provides interpretable intermediate artifacts, so failure can be diagnosed before training a policy. It also avoids requiring a large dataset of teleoperated G1 demonstrations for every new scene and instruction.

The same decomposition also makes limitations explicit. Generated videos are not guaranteed to be physically valid. They may show foot contacts that are visually plausible but dynamically impossible, or they may hide important geometry behind the robot. Monocular reconstruction remains fragile under occlusion, reflective surfaces, and viewpoint-dependent hallucination. A per-video policy-training loop may be too slow for interactive use. Finally, robot pose estimation for a generated G1 is an open problem: the input is not a real image with sensor noise, but a generated image with model-specific artifacts.

The current prototype is one step in this program. It demonstrates the scene-reconstruction side of the pipeline and leaves robot pose estimation, reference tracking, and physical deployment as active next stages. Accordingly, we focus claims on the implemented conversion from generated monocular video to metric, gravity-aligned scene geometry, and treat the complete chain from generated video to recovered reference, simulated tracking, and hardware execution as the next evaluation target.

Future work will focus on completing the G1 pose-estimation module, training a reference-tracking policy in the reconstructed mesh, and testing whether the resulting references are physically trackable. Beyond completing this chain, the same framing opens several directions: automatic scale and

gravity recovery, contact-aware trajectory repair, dynamic scenes with moving objects, manipulation and loco-manipulation, generation directly in 3D or latent world representations, and zero-shot or universal trackers that can execute recovered references without training a new policy for each generated video.

7 Conclusion

We presented GVHC, a pipeline for using generative video as an intermediate representation on the path toward Unitree G1 humanoid control. The system starts from language and a single image, generates a short task video, and converts that video into explicit geometric structure for simulation. The implemented prototype reconstructs a metric, gravity-aligned static scene mesh from a generated monocular video using segmentation, point-map prediction, robust static registration, and Poisson meshing. The robot pose, policy training, and real-world deployment stages are in progress.

The broader claim is that generated videos can be useful when their recoverable structure is made explicit: approximate motion, spatial layout, and task-relevant geometry. This turns a video model into a source of imagined demonstrations, while still routing execution through geometry, motion tracking, and physics. For humanoid locomotion in static scenes, GVHC offers a compact and extensible path from language-conditioned visual generation toward physically grounded control.

References

[1] Google DeepMind. Veo 3.1, 2025. URL https://deepmind.google/models/veo/. Accessed 2026-05-26.

[2] R. Geirhos et al. Video Models Are Zero-Shot Learners and Reasoners. *arXiv preprint arXiv:2509.20328*, 2025.

[3] Meta AI. Segment Anything Model 3, 2025. URL https://ai.meta.com/sam3/. Accessed 2026-05-26.

[4] R. Wang, S. Xu, Y. Dong, Y. Deng, J. Xiang, Z. Lv, G. Sun, X. Tong, and J. Yang. MoGe-2: Accurate monocular geometry with metric scale and sharp details. *arXiv preprint arXiv:2507.02546*, 2025.

[5] T.-X. Xu, X. Gao, W. Hu, X. Li, S.-H. Zhang, and Y. Shan. GeometryCrafter: Consistent geometry estimation for open-world videos with diffusion priors. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025.

[6] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Eurographics Symposium on Geometry Processing*, 2006.

[7] M. Kazhdan and H. Hoppe. Screened Poisson surface reconstruction. *ACM Transactions on Graphics*, 32(3), 2013.

[8] Y. Du, M. Yang, B. Dai, H. Dai, O. Nachum, J. B. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning universal policies via text-guided video generation. In *Proceedings of the International Conference on Machine Learning*, 2023.

[9] Y. Du, M. Yang, P. Florence, F. Xia, A. Wahid, B. Ichter, P. Sermanet, T. Yu, P. Abbeel, J. B. Tenenbaum, L. P. Kaelbling, A. Zeng, and J. Tompson. Video language planning. In *Proceedings of the International Conference on Learning Representations*, 2024.

[10] J. Jang, S. Ye, Z. Lin, J. Xiang, J. Bjorck, Y. Fang, F. Hu, S. Huang, K. Kundalia, Y.-C. Lin, L. Magne, A. Mandlekar, A. Narayan, Y. L. Tan, G. Wang, J. Wang, Q. Wang, Y. Xu, X. Zeng, K. Zheng, R. Zheng, M.-Y. Liu, L. Zettlemoyer, D. Fox, J. Kautz, S. Reed, Y. Zhu, and L. Fan. DreamGen: Unlocking generalization in robot learning through neural trajectories. *arXiv preprint arXiv:2505.12705*, 2025.

[11] A. Allshire, H. Choi, J. Zhang, D. McAllister, A. Zhang, C. M. Kim, T. Darrell, P. Abbeel, J. Malik, and A. Kanazawa. VideoMimic: Visual imitation enables contextual humanoid control. *arXiv preprint arXiv:2505.03729*, 2025.

[12] T. He, J. Gao, W. Xiao, Y. Zhang, Z. Wang, J. Wang, Z. Luo, G. He, N. Sobanbabu, C. Pan, Z. Yi, G. Qu, K. Kitani, J. Hodgins, L. Fan, Y. Zhu, C. Liu, and G. Shi. ASAP: Aligning simulation and real-world physics for learning agile humanoid whole-body skills. *arXiv preprint arXiv:2502.01143*, 2025.

[13] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn. HumanPlus: Humanoid shadowing and imitation from humans. In *Proceedings of the Conference on Robot Learning*, 2024.

[14] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[15] J. Zhang et al. MonST3R: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2025.

[16] T. E. Lee, J. Tremblay, T. To, J. Cheng, T. Mosier, O. Kroemer, D. Fox, and S. Birchfield. Camera-to-robot pose estimation from a single image. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2020.

[17] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. Single-view robot pose and joint angle estimation via render and compare. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.