ROSE: Reconstructing Objects, Scenes, and Trajectories from Casual Videos for Robotic Manipulation

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper, we build a real-to-sim-to-real (Real2Sim2Real) system for robot manipulation policy learning from casual human videos. We propose a new framework, ROSE, that directly leverages casual videos to reconstruct simulator-ready assets, including objects, scenes, and object trajectories, for training manipulation policies with reinforcement learning in the simulation. Unlike existing real-to-sim pipelines that rely on specialized equipment or time-consuming and labor-intensive human annotation, our pipeline is equipment-agnostic and fully automated, facilitating data collection scalability. From casual monocular videos, ROSE enables the direct reconstruction of metric-scale scenes, objects, and object trajectories with physics information in the same gravity-calibrated coordinate for robotic data collection in the simulator. With ROSE, we curate a dataset of simulator-ready scenes from casual videos from our own capture and the Internet, and create a benchmark for real-to-sim evaluation. Across a diverse suite of manipulation tasks, ROSE outperforms the existing baselines, laying the groundwork for scalable robotic data collection and achieving efficient Real2Sim2Real deployment.

1 Introduction

Learning complex manipulation skills from human demonstrations is a long-standing goal in robotics. Casual human videos offer a vast and diverse source of demonstrations, but translating this unstructured visual data into executable robot policies is a significant challenge. Since replicating these scenes in the real world for direct imitation is often impractical or unsafe, a more scalable approach is needed.

This paper develops a real-to-sim-to-real system that bridges this gap, enabling robust policy learning by first reconstructing the essence of a human demonstration within a physically-realistic simulator. Three features need be *simultaneously* present in one *gravity-aligned, metric world frame*: (G) *collidable geometry* for both scene and objects; (P) *physics plausibility* so that contact, scale, and gravity are sensible in simulation (coarse priors over material/density belong here, but exact identification is not the point); and (M) *executable motion*—a time-aligned 6-DoF object trajectory the robot can imitate, replay, or condition on. The minimal sufficient assets that let a robot both plan and act are this G/P/M triad, coherently aligned in world coordinates.

Previous works have explored reconstructing robotics manipulation data from human videos. However, we argue that *four* elements based on the G/P/M triad need to co-occur for turning casual video into robotic data as a scalable data engine—**casual video**, **object trajectory**, **scene**, and **object**. It surfaces where representative methods in Tab. 1 leave gaps. (1) *Being Able to Reconstruct from Casual Video.* Internet-scale demonstrations are predominantly casual (narrow baseline, dynamics).

Table 1: **Comparison with existing real-to-sim pipelines**. Scene mesh: 3D collision mesh of the scene. Object Mesh: 3D collision mesh of objects. Object Traj.: The 6-DoF pose of objects to be manipulated. Gravity Dir: The gravity direction of the reconstructed scene and objects. Metric Scale: If the reconstructed scene is in metric space (cm). World Coord.: If the reconstructed scene is in the world coordinate. Automation: It is a fully automated pipeline or requires human annotations (*e.g.*, RialTo [57] needs expert human annotation using GUI tools). O: Unknown.

	Characteristics							
Method	Scene Mesh	Object Mesh	Object Traj.	Gravity Dir.	Metric Scale	World Coord.	Automation	**Input / Platform**
RialTo [57]	✓	✓	✓	✓	✓	✓	✗	RGB-Video
Video2Policy [71]	✗	✓	✓	✗	✓	✗	✓	MV-imgs / LiDAR
RL-GSBridge [65]	✗	✓	✗	✗	✗	✓	✗	MV-imgs
SplatSim [46]	✓	✓	✓	✗	✗	✓	✗	RGB-D
ReBot [10]	✗	✗	✓	✗	✓	✓	✓	Video / Mesh
Digital Cousins [7]	O	O	✗	✗	O	✓	✓	Image
Chen et al. [4]	✗	✗	✗	✗	✗	✗	✓	Mesh / Trajectory
URDFormer [5]	✓	✓	✗	✗	✗	✓	✓	MV-imgs
Ditto In the House [19]	✓	✓	✗	✗	✗	✓	O	Image / Interaction
Ours	✓	✓	✓	✓	✓	✓	✓	RGB-Video

GS-based pipelines such as [46, 57, 65] are typically tuned for multi-view, stable captures and are sensitive to dynamics or camera-trajectory violations, which limits robustness under single-take narrow-baseline inputs; several other entries rely on multi-view or controlled capture [4, 5, 7, 10, 19]. (2) *Object trajectory.* A world-aligned 6-DoF object trajectory provides task-resolution signals that planners and policies can directly reuse; video-conditioned control and end-to-end policy learning confirm the value of motion cues [21, 71], while methods emphasizing 3D assets or interaction without exporting an executable, world-frame object trajectory offer less leverage for planning [5, 19, 57]. (3) *Scene.* Object motion is contextual: contact feasibility, clearances, and gravity alignment are defined with respect to the scene; pipelines that do not reconstruct a collidable, metric-scale, gravity-aligned scene (e.g., learning directly from videos without scene geometry [71]) provide limited support for validating contacts or credibly replaying motion. (4) *Object.* Interaction requires an *interactable* object; modern simulators assume well-behaved meshes for stable contact, so policies validated without object geometry risk overfitting to appearance rather than contact-rich behavior. Taken together, partial solutions in Tab. 1 (e.g., scene/object without motion, motion without scene, or assets lacking world/scale/gravity alignment) are valuable components but harder to use as a generalizable engine. Our aim is to deliver *simulator-ready* scene/object meshes and an executable, world-aligned object trajectory from casual video so that downstream planners and policies can both plan and act.

Our thesis is that casual, narrow-baseline videos can be turned into such assets at scale *if* two design choices are made early and enforced end-to-end: first, *unify* camera, metric scale, and gravity so that geometry and motion live in a single world frame throughout; second, treat reconstruction as a high-throughput generator and guard it with a re-rendering consistency gate. We use SSIM with geometric/physical sanity checks to convert long-tail failures into discardable samples before errors cascade. In addition, we treat *physics as plausibility*: we incorporate coarse, category-conditioned priors (e.g., mass estimated by VLM) into the simulation setup to avoid obviously non-physical interactions without over-promising fine identification [34, 47, 62].

Concretely, from a single casual video we (i) recover cameras, metric scale, gravity and physics information; (ii) produce *collidable* scene and object meshes suitable for standard simulators; (iii) estimate an *object-centric 6-DoF trajectory* consistent with that world frame; and (iv) enforce a *post-filtering gate* via differentiable re-rendering and SSIM, complemented by geometric and physical checks. The result is a *simulator-ready* bundle with geometry, physics plausibility and motion produced automatically from casual video.

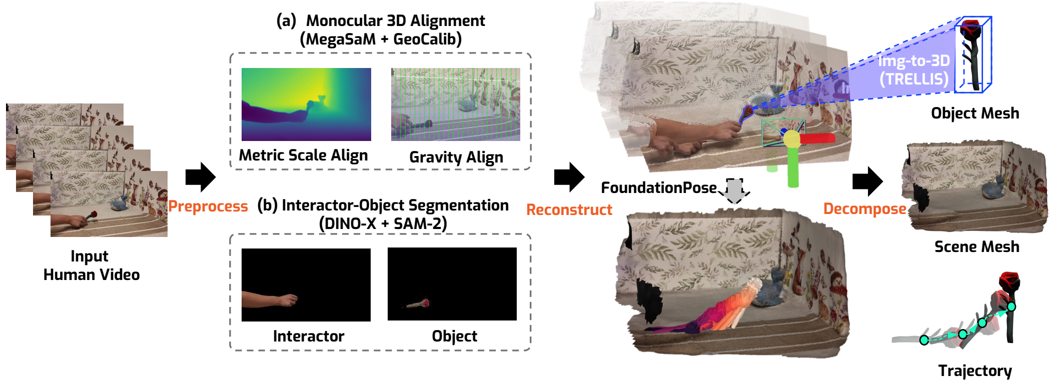

Figure 1: **ROSE Real2Sim pipeline illustration.** (a) We leverage MegaSaM[31] and GeoCalib[58] to reconstruct scene point cloud in the metric-scale and gravity-align world coordinates. (b) We further use SAM-2[48] and DINO-X[49] to detect and track interactor and object mask from videos.

Given only a single RGB camera moving through a real scene, ROSE automatically builds an interactive 3D simulation of that scene, reconstructing the geometry, appearance, and physics information of objects and surfaces. The resulting simulation (a "digital twin" of the scene) can be used to train and evaluate manipulation strategies in a safe and scalable manner. By eliminating much of the manual effort required to create detailed simulated environments, ROSE aims to enable robots to learn and test manipulation policies in faithful virtual replicas of real-world settings, then execute them reliably in the physical world. We collect a large-scale dataset comprising diverse scenes, objects, trajectories, and physically plausible robotic actions for task completion. The dataset includes more than 30 scenes, 50 objects, 600 trajectories, and 3,500 robot action samples.

2 Related Work

2.1 Sim-to-Real RL Policy Transfer

Training robot policies with RL in simulation, followed by a sim-to-real (Sim2Real) policy transfer, has become one of the most successful robot learning strategies in wide applications, such as locomotion [18, 25, 26, 32, 55], loco-manipulation [11, 12, 17, 51], dexterous manipulation [16, 45], *etc*. One advantage of such Sim2Real RL training lies in the low-cost, safe, and more potential in improving generalization through domain / dynamic randosmizations [41, 56], making it a widely adopted alternative to collecting real-world data that is typically time-consuming and labor-intensive. However, such a low-cost and safe simulation training alternative may bring a Sim2Real gap that makes it hard for the Sim2Real policy transfer. To address this issue, a lot of works have been proposed to mitigate the gap, *e.g.*, curriculum learning of Sim2Real constraints [18, 27, 33, 52], teacher-student distillation of privileged information like object states or environment extrinsics [25, 27, 43], 3D awareness [15, 23, 39, 54, 72], and perception augmentation / randomization [1, 2, 6, 9, 35, 50].

2.2 Real-to-Sim Dynamic Scene and Object Transfer

Recently, a lot of efforts in 3D vision have been devoted to creating simulated twins of the real-world scenes / objects from 2D videos [22, 30, 38], which is critical in enriching operating environments when training robot policies in simulation. Generally, transferring real-world scene videos to the 3D simulation that is useful for robot learning involves three key components: i) 3D scene geometry, ii) 3D object geometry, and iii) object dynamics, which requires two key techniques as follows.

Dynamic 3D Scene Reconstruction from 2D focuses on recovering the appearance and geometry of scenes from 2D images or videos. Earlier methods [28, 30, 59, 64, 70] typically rely on dense multiview capture and require significant computational resources to reconstruct dynamic scenes, often using NeRF-based [37] or 3D Gaussian Splatting [24] representations that evolve over time. More recently, with advances in deep multiview stereo [29, 61] and monocular depth estimation [42, 69], a new line of work has emerged that better captures the geometry of dynamic scenes from casual inputs. Notably, approaches such as MegaSaM [31], MonST3R [73], and CUT3R [60] demonstrate robust and efficient dynamic 3D reconstruction from casually captured monocular videos. These methods mark a significant step towards scalable, large-scale scene reconstruction and asset creation for downstream applications like robotics.

3D Object Dynamics from 2D provides the object-level kinematic dynamics encoded as object spatial translation and orientation in 3D, offering valuable priors that help both traditional motion planning methods and learning-based approaches. To capture such object dynamics, various object representations have been used as the policy tracking goal. For example, Bharadhwaj et al. [2] propose to use object and hand segmentation as proxy information, followed by a segmentation image conditioned policy that achieves better generalization.

Meanwhile, some works utilize the point-level flow map of objects or images as the point tracking objective and achieve great progress [3, 8, 13, 67]. Different from relying on such proxy representations, our approach directly collects 6DoF trajectories through pose estimation [63], offering a scalable and efficient solution for acquiring high-quality motion data. Notably, a concurrent work, Video2Policy [71], also proposes to use 6DoF object trajectories as object dynamics. However, Video2Policy only reconstructs the object states and places objects on the same canonical tabletop in a specific robot frame. In contrast, our approach transfers both the dynamic scenes and the objects in the world coordinate, where the world frame reconstruction helps SLAM-based scene reconstruction [31].

3 ROSE:Reconstructing Object, Scene, and Trajectory

3.1 Object Reconstruction

Object Grounding. As shown in Fig. 1, given a target object label from user input or LLM inference, we scan the video frame-by-frame until the object is first detected by DINO-X [49]. The detected bounding box is passed to SAM 2 [48] to obtain the initial target object mask $\mathbf{M}^{\text{init}}$. SAM 2 then propagates this mask through the remainder of the sequence, yielding per-frame object masks $\{\mathbf{M}_t\}_{t=1}^N$.

Object Mesh Reconstruction. Using the segmented masks, we leverage TRELLIS [66] to reconstruct the 3D object mesh $\mathcal{M}^{\text{obj}}$, which provides 3D reconstruction pipelines from both single image and multiview images. Since most manipulation videos are filmed from a single viewpoint, we select the initial mask $\mathbf{M}^{\text{init}}$ to reconstruct $\mathcal{M}^{\text{obj}}$. For highly occluded or feature unclear situation, we would use masks from multiple non-occuluded views to reconstruct.

3.2 Scene Reconstruction

Scene Point Cloud Reconstruction. For every video frame $\mathbf{I}_t$, MegaSaM [31] provides the camera intrinsics $\mathbf{K}_t$, the camera pose $\mathbf{G}_t = [\mathbf{R}_t|\mathbf{t}_t]$, and a relative depth map $\mathbf{D}_t^{\text{rel}}$. We feed $\mathbf{I}_t$ to UniDepth [42] to obtain an absolute depth estimate $\mathbf{D}_t^{\text{abs}}$. A global scale factor $\hat{\alpha}$ and offset $\hat{\beta}$ align $\mathbf{D}_t^{\text{rel}}$ to metric depth $\mathbf{D}_t^{\text{align}}$. Each pixel is back-projected using $\mathbf{K}_t$, $\mathbf{G}_t$ and $\mathbf{D}_t^{\text{align}}$ to obtain its corresponding 3D point, which we accumulate into a raw scene point cloud $\mathcal{P}$. Then we apply GeoCalib [58] onto the first frame and obtain a gravity-align transformation $\mathbf{P}^{\text{gravity}}$. Then we apply $\mathbf{P}^{\text{gravity}}$ to each of the following frame to ensure the scene is under the gravity-aligned coordinate.

Scene Mesh Reconstruction. The sparse, hole-ridden point cloud $\mathcal{P}$ yielded by the previous stage is first densified with Neural Kernel Surface Reconstruction (NKSR) [20]; its *detail* hyper-parameter is tuned to close gaps while preserving fine geometry. To satisfy simulator requirements, namely orientability, 2-manifoldness, and self-intersection freedom, we subsequently apply an Alpha Wrapping procedure [44], producing a watertight, validity-guaranteed surface. Finally, color is restored by a point-to-vertex transfer: each mesh vertex inherits the distance-weighted average RGB of its three nearest neighbors in the processed point cloud, yielding a textured, simulation-ready scene mesh $\mathcal{M}^{\text{scene}}$.

3.3 Trajectory Reconstruction

Improved Foundation Pose. Given a set of segmentation masks $\{\mathbf{M}_t\}_{t=1}^N$, we employ Foundation-Pose [63] in a model-based setting to estimate the object's 6-DoF pose $\mathbf{P}^{\text{obj}}$. The estimator takes as input the RGB image $\mathbf{I}$, the reconstructed object mesh $\mathcal{M}^{\text{obj}}$ from Sec. 3.1, the camera intrinsics matrix $\mathbf{K}$, and the aligned depth map $\mathbf{D}^{\text{align}}$ from Sec. 3.2. Since FoundationPose assumes metric consistency between depth and mesh, we introduce a scale-alignment procedure.

Specifically, we backproject pixels (u, v) inside $\mathbf{M}^{\text{init}}$ to 3D points $\mathbf{p}(u, v) \in \mathbb{R}^3$ in camera coordinates using $\mathbf{D}^{\text{align}}$ and $\mathbf{K}$. We then compute the maximum pairwise distance among these points,

$$\mathbf{d}^{\text{image}} = \max_{(u_1,v_1),(u_2,v_2)\in\mathbf{M}^{\text{init}}} \|\mathbf{p}(u_1, v_1) - \mathbf{p}(u_2, v_2)\| \tag{1}$$

For the mesh, we compute the maximum vertex-to-vertex distance $\mathbf{d}^{\text{mesh}}$,

$$\mathbf{d}^{\text{mesh}} = \max_{\mathbf{x}_a,\mathbf{x}_b\in\mathcal{M}^{\text{obj}}} \|\mathbf{x}_a - \mathbf{x}_b\|. \tag{2}$$

These yield an initial scale estimate $\rho_0 = \mathbf{d}^{\text{image}}/\mathbf{d}^{\text{mesh}}$.

To account for noise in depth, intrinsics, and occlusions, we refine the scale by a discrete search

$$\rho \in \big[\rho_0/\alpha,\ \rho_0\alpha\big] \quad \text{with step size } s.$$

For each candidate scale ρ, we scale the mesh $\mathcal{M}^{\text{obj}}$, run FoundationPose to obtain the pose $\mathbf{P}^{\text{obj}}(\rho)$, render the corresponding silhouette $\hat{\mathbf{M}}(\rho)$, and select the ρ^* that minimizes the IoU loss with the ground-truth mask $\mathbf{M}^{\text{init}}$:

$$L_{\text{IoU}}(\rho) = 1 - \frac{\sum_{i\in\Omega} \hat{m}_i(\rho)\, m_i}{\sum_{i\in\Omega}\big(\hat{m}_i(\rho) + m_i - \hat{m}_i(\rho)\, m_i\big)}. \tag{3}$$

Here Ω denotes the image domain, $m_i \in \{0, 1\}$ is the ground-truth mask value, and $\hat{m}_i(\rho) \in [0, 1]$ is the rendered mask value at pixel i. We take the final pose as $\mathbf{P}^{\text{obj}} = \mathbf{P}^{\text{obj}}(\rho^*)$ and use it to initialize pose tracking, yielding poses for subsequent frames.

3.4 Post Filtering System

To ensure our pipeline maintains a high level of quality we run each result through a filtering system. In order to holistically evaluate the quality of our results, we combine our reconstructed scene, object, trajectory, and camera poses and render them into a reconstructed video to compare with the original video. This formulation enables our filtering system to remove results which may have malformed object meshes, incorrectly textured scenes, or inaccuracies in trajectory and pose reconstruction as well as filter based on the accumulated error from each of these potential sources over time.

To ensure there is temporally consistent accuracy between our pipeline's reconstruction and original video, our filtering system places the object back into the scene following the reconstructed trajectory from each time step of our pipeline's result. For each object position we render a frame from the corresponding reconstructed camera pose. Upon rendering each of the reconstructed frames, f^{recon}, we compute the SSIM with each of the video's original frames, f^{gt}. We average across all the frames and finally filter results based on an established threshold from previous success and failure cases.

$$\text{SSIM}_{\text{avg}} = \frac{1}{N}\sum_{t=1}^{N} \text{SSIM}(f_t^{\text{recon}}, f_t^{\text{gt}}) \tag{4}$$

$$\text{success} = \begin{cases} 1, & \text{if SSIM}_{\text{avg}} > \text{SSIM}_{\text{thresh}} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

With this filtering system, our pipeline gains the capability to reconstruct scenes from large numbers of casual videos, offering scalability while ensuring that reconstruction quality is maintained.

3.5 Robot Action Collection

Building on the object trajectories reconstructed by the pipeline described above, we further explain how we collect robotic action data to enable the object to follow the trajectory and complete the task. With the reconstructed scene and object, we first load them into the simulator. Given the object's motion, our goal is to control the robot to interact appropriately with the object and guide it along the desired trajectory. We primarily utilize two baseline approaches for diverse robotic action collection: motion planning-based and reinforcement learning-based methods.

Motion Planning. For the motion planning-based algorithm, we first predict an appropriate grasping pose for the object. Once a stable grasp is achieved, the robot follows the object's trajectory using end-effector control based on cuRobo[53]. If the object remains stable and the trajectory is successfully followed, a data sample is considered successfully collected. For this method, we only consider the parallel-jaw gripper setting. In detail, we use GSNet [40] to predict grasp poses based on the point cloud generated in the simulation. After executing a planned trajectory to successfully grasp the object, the robot then follows the trajectory obtained from our vision pipeline to collect valid data.

Reinforcement Learning. Although the motion planning-based method is efficient and easy to implement, it is not sufficient for all scenarios. For example, when using high-dimensional robotic hands, as opposed to simple parallel-jaw grippers, predicting an appropriate grasping pose becomes significantly more challenging. In such cases, reinforcement learning (RL) allows the robot to explore and learn effective grasping strategies to complete the task.

Our RL baseline consists of two stages: object grasping and object manipulation. In the first stage, we design a reward function composed of three terms: a reaching reward r_{reach}, a grasping reward r_{grasp}. In the second stage, we follow the object trajectory generated by our previous pipeline to complete the task. To achieve this, we use CuRobo to control the end-effector and track the trajectory accurately.

It is also worth noting that we explored an end-to-end RL approach without the two-stage setting. While we carefully designed a reward function for trajectory following, we found that it was difficult for the policy to accurately replicate the generated motion, particularly in cases involving complex rotations. This limitation arises from the inherent nature of RL: since learning relies heavily on exploration, it is challenging for a policy to acquire precise trajectory-following behavior, especially when the robot is simultaneously required to grasp and manipulate the object.

3.6 Sim-to-real Transfer

With action data collected, we further train a model for sim-to-real transfer. A key advantage of our vision pipeline is its ability to generate high-quality, simulation-ready scene and object meshes, along with corresponding object trajectories. This enables fast and accurate robotic data collection in simulation. Using this data, we can leverage a high-quality renderer to produce realistic visual datasets. This allows us to train a vision-based robotic model capable of directly transferring to real-world scenarios.

4 Experiments

4.1 Experiment Setup

Our model is able to collect robotic data from diverse datasets from various sources, including outdoor, indoor environments. We benchmarked our real-to-sim method in RoboVerse[14] simulation environment and validated it in both simulation and real-world settings using the Franka arm and Unitree G1 humanoid robots.

4.2 Benchmark Construction

We construct a new benchmark to evaluate the fidelity of real-to-sim-to-real pipeline scene reconstructions from casual monocular videos as shown in Tab. 2. Because existing metrics treat scene layout, object shape, and motion separately, our benchmark fuses them into one holistic evaluation. It provides five simulated environments with full ground-truth geometry, appearance, and trajectories, plus a casually captured video that serves as the pipeline's input. Evaluation uses four metrics: per-frame Chamfer distance between scene point clouds, Chamfer distance for object geometry, and APE/RPE (translation and rotation) for object trajectories. Scores are averaged across frames to yield stable measures. Together, these metrics reveal how well a method recovers both the static environment and the dynamics of the objects within it.

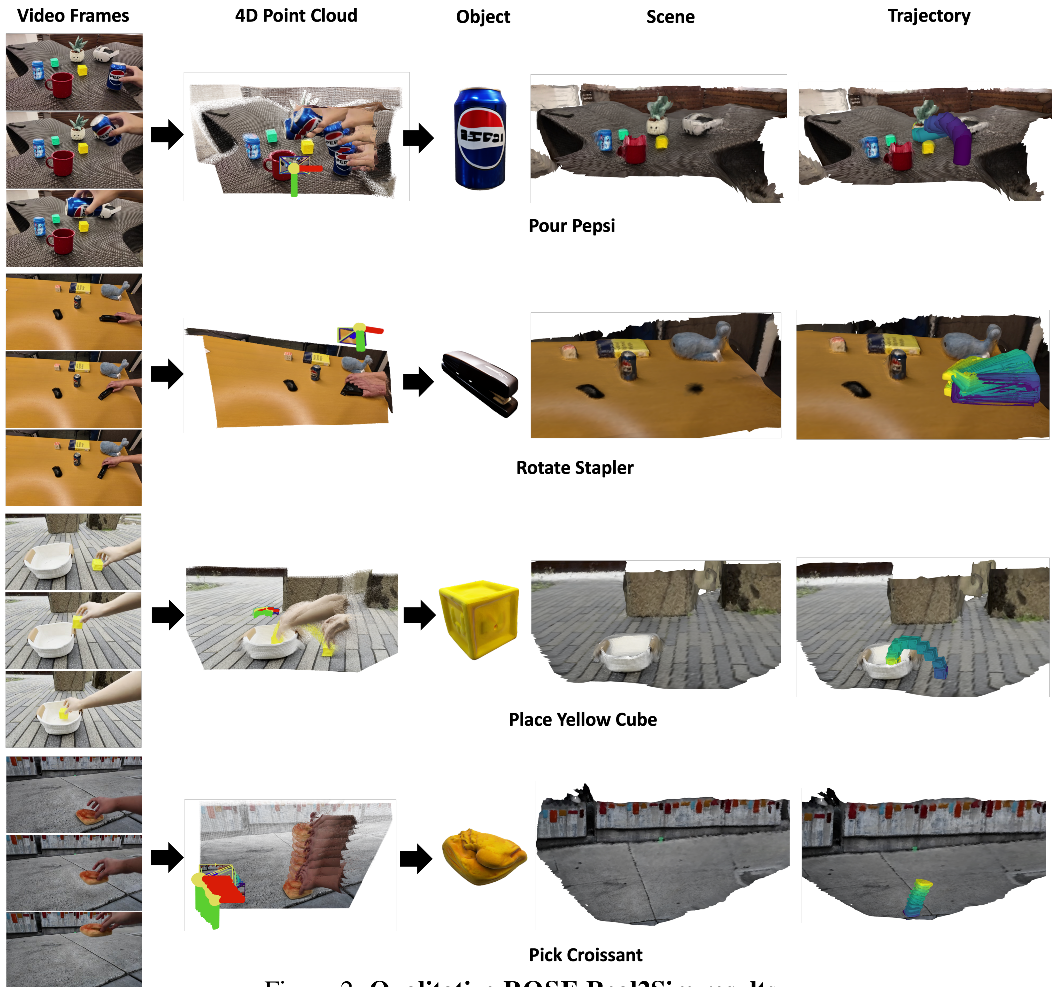

Figure 2: **Qualitative ROSE Real2Sim results.**

Task	Avg. Scene Chamfer Dist.	Object Chamfer Dist.	Translation APE	Rotation RPE	Translation RPE
Unstack	0.6211	0.02158	0.003242	3.724	0.001649
Place	0.6945	0.01060	0.02629	3.804	0.02269
Lift	0.6696	0.02786	0.02208	9.065	0.004374
Push	0.7513	0.01516	0.01086	4.229	0.002170
Rotate	0.6513	0.01394	0.008418	3.508	0.003301
Average	0.6776	0.01782	0.01418	4.866	0.006837

Table 2: **Benchmark comparison across tasks.** ROSE's performance metrics from our benchmark. Avg. Chamfer distance is computed for scene reconstructions, while object metrics include Chamfer distance, Absolute Pose Error (APE), and Relative Pose Error (RPE).

4.3 Data Generation Time and Qualitative Results

With post filtering system we are able to reconstruct scenes from casual videos while maintaining the quality of our results. We present qualitative results on Fig. 2, demonstrating how our pipeline reconstructs geometrically accurate scene, object and object trajectory from casual videos to enable policy training. Additionally we compare the runtime of our pipeline against the runtime of the leading baseline which can be seen in Tab. 3. ROSE reconstructs environments and trajectory data around **8x faster** than the baseline on while remaining accuracy on geometry.

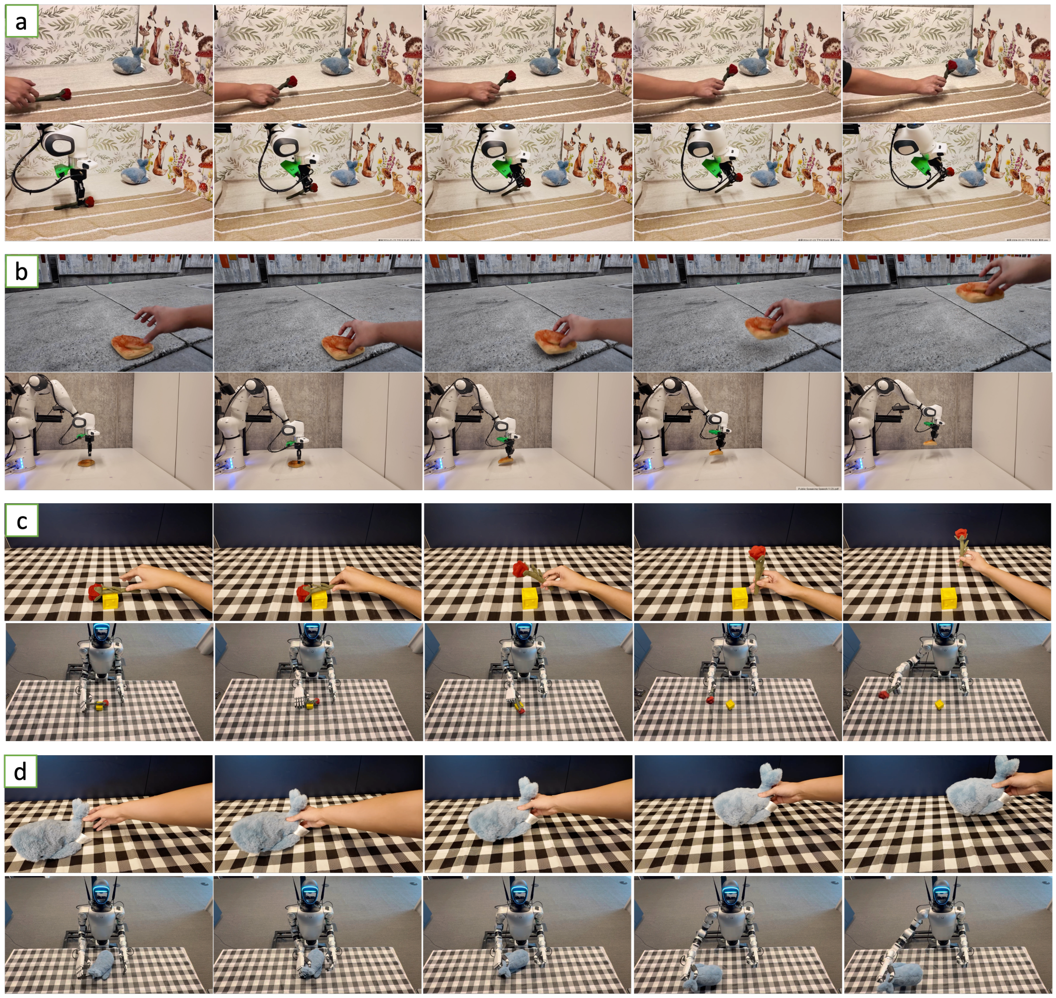

Figure 3: **Qualitative ROSE real-world results.**

Task Name	ROSE (Ours)		Improved V2P	
	Recon Time ↓	SSIM ↑	Recon Time ↓	SSIM ↑
Triangle Move Mouse	**8m46s**	**0.803**	72m39s	0.746
Circle Move Mouse	**8m28s**	**0.789**	71m45s	0.734
Flip Magic Cube	**7m57s**	**0.718**	70m37s	0.598
Rotate Stapler	**8m39s**	**0.715**	83m01s	0.582
Pour Pepsi	**9m22s**	**0.713**	77m58s	0.632

Table 3: **Pipeline Runtime and Reconstruction Quality.** Comparison of ROSE and Video2Policy (V2P) with 3DGS-based scene reconstruction runtimes and SSIM values.

4.4 Robotic Dataset Collection

Leveraging our scene, object, and trajectory reconstruction results, along with our robotic data collection pipeline, we construct a robotic manipulation dataset from monocular video. In the end, we collect **3.5k** valid robotic datasets with diverse task settings and environment variation.

Simulation Environment Setup. Based on RoboVerse [14], we develop a pipeline for generating simulation environments. We use a standardized configuration file to process scene layouts and object meshes. After loading the target robot into the simulation, we perform unit tests to ensure proper setup and collision-free initialization. We then follow the data collection pipelines to gather robotic manipulation data.

Manipulation Benchmark in Simulation. We establish a simulation benchmark to evaluate the performance of different robotic data collection methods. Specifically, we compare our proposed motion-planning-based approach and a two-stage reinforcement learning (RL) method against an end-to-end RL baseline. Our results show that the motion planning-based and two-stage RL methods perform differently across various settings—each demonstrating strengths in different scenarios. Comparing with strong baselines, including the concurrent work Video2Policy [**?**], Our method achieves the best performance on three out of four tasks as well as on the average score as shown in Tab. 4.

Method	PickPepsi	StackBlock	PlaceBowl	MoveTriangle	Average
End-to-End RL	**1.00**	0.00	**1.00**	0.00	0.50
Video2Policy [71]	0.00	0.00	0.40	0.00	0.10
Ours (Motion Planning)	0.80	**1.00**	0.40	0.80	0.75
Ours (Two-stage RL)	**1.00**	0.60	**1.00**	**1.00**	**0.90**

Table 4: **Task completion rate in simulation.**

4.5 Sim-to-Real Transfer

To validate the usefulness of our collected data, we conduct experiments to demonstrate the effectiveness of both the dataset and the trained policy.

Zero-shot Robotic Manipulation and Data Collection. We evaluate our collected robotic data and data collection pipeline in real-world settings. Specifically, we deploy the motion-planning-based method as shown at c and d row in Fig 3 in a physical environment to assess its capability for zero-shot data collection and task execution using only a single demonstration. We test the data collection system across 13 different scenarios, achieving success in 11 of them—resulting in an **84.6%** success. The failure is primarily due to incorrect grasp poses and joint limit violations during motion planning.

Policy Sim-to-Real Transfer. We further train an RGB-based policy in simulation and demonstrate that, using the assets generated by our real-to-sim pipeline, the action data collected in simulation, and high-quality rendering based on RoboVerse [14], the resulting policy can zero-shot generalize to the real world.

5 Conclusion

We presente ROSE, an end-to-end, equipment-agnostic Real2Sim pipeline that lifts casual monocular videos into simulator-ready assets: metric-scale, gravity-aligned scene reconstructions, watertight textured meshes that meet simulator validity constraints, and consistent 6-DoF object trajectories in a unified world frame. The system combines robust geometric recovery with a post-hoc filtering stage that enforces temporal consistency, yielding assets that can be consumed directly by both motion-planning and learning-based controllers. On top of this vision stack, we standardized the handoff to robotics—curating a benchmark that evaluates scene, object, and trajectory fidelity jointly, and building data-collection routines that translate demonstrations into scalable simulation rollouts.

Empirically, ROSE recovers geometry and dynamics with strong fidelity across diverse manipulation tasks and outperforms prior Real2Sim baselines in simulation, while also enabling zero-shot transfer on real robots. Together with the curated assets and 3D reconstructions, these results indicate that high-quality manipulation data can be collected from casual videos at scale, reducing manual environment authoring and separating risky exploration from the real world. We view ROSE as a step toward video-driven Real2Sim2Real at population scal, where casual videos become a training substrate for policies. It standardizes the interface between perception, asset creation, and policy learning, and opens a path to richer, safer, and more diverse robotic data collection.

6 Limitations

Our current scope is restricted to rigid scenes and objects. Articulation, deformable materials, and fluid-like dynamics are not modeled, and we do not yet reason about contact compliance or material parameters beyond simple frictional settings. The object mesh is typically reconstructed from limited viewpoints; under severe occlusion, low texture, specularity, or translucency, geometry and pose tracking may degrade. As with any monocular pipeline, metric scale and intrinsics are estimated rather than measured, making ROSE susceptible to residual scale or gravity misalignment in challenging conditions.

On the control side, our data-collection stack emphasizes parallel-jaw grasping and a two-stage policy structure; purely end-to-end trajectory following proved fragile for complex rotations, and we do not yet leverage force/tactile feedback. Extending ROSE to articulated and deformable objects, multi-object interactions, richer physics priors, tactile sensing, broader robot platforms, and physically informed filtering is an important next step toward making casual videos a dependable substrate for policy learning at scale.

References

[1] Ananye Agarwal, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Legged locomotion in challenging terrains using egocentric vision. In Karen Liu, Dana Kulic, and Jeffrey Ichnowski, editors, *CoRL*, Proceedings of Machine Learning Research. PMLR, 2022.

[2] Homanga Bharadhwaj, Abhinav Gupta, Vikash Kumar, and Shubham Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. In *ICRA*. IEEE, 2024.

[3] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *ECCV*. Springer, 2024.

[4] Yuanpei Chen, Chen Wang, Yaodong Yang, and Karen Liu. Object-centric dexterous manipulation from human motion data. In *CoRL*.

[5] Zoey Chen, Aaron Walsman, Marius Memmel, Kaichun Mo, Alex Fang, Karthikeya Vemuri, Alan Wu, Dieter Fox, and Abhishek Gupta. Urdformer: A pipeline for constructing articulated simulation environments from real-world images, 2024.

[6] Zoey Qiuyu Chen, Shosuke C. Kiami, Abhishek Gupta, and Vikash Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. In Kostas E. Bekris, Kris Hauser, Sylvia L. Herbert, and Jingjin Yu, editors, *Robotics: Science and Systems XIX, Daegu, Republic of Korea, July 10-14, 2023*, 2023.

[7] Tianyuan Dai, Josiah Wong, Yunfan Jiang, Chen Wang, Cem Gokmen, Ruohan Zhang, Jiajun Wu, and Li Fei-Fei. Automated creation of digital cousins for robust policy learning. In *CoRL*, 2024.

[8] Bardienus P Duisterhof, Zhao Mandi, Yunchao Yao, Jia-Wei Liu, Jenny Seidenschwarz, Mike Zheng Shou, Deva Ramanan, Shuran Song, Stan Birchfield, Bowen Wen, et al. Deformgs: Scene flow in highly deformable scenes for deformable object manipulation. *arXiv preprint arXiv:2312.00583*, 2023.

[9] Hongjie Fang, Hao-Shu Fang, Sheng Xu, and Cewu Lu. Transcg: A large-scale real-world dataset for transparent object depth completion and a grasping baseline. *IEEE Robotics and Automation Letters*, (3), 2022.

[10] Yu Fang, Yue Yang, Xinghao Zhu, Kaiyuan Zheng, Gedas Bertasius, Daniel Szafir, and Mingyu Ding. Rebot: Scaling robot learning with real-to-sim-to-real robotic video synthesis. *arXiv preprint arXiv:2503.14526*, 2025.

[11] Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion. In Karen Liu, Dana Kulic, and Jeffrey Ichnowski, editors, *CoRL*, Proceedings of Machine Learning Research. PMLR, 2022.

[12] Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetzstein, and Chelsea Finn. Humanplus: Humanoid shadowing and imitation from humans. In *CoRL*, 2024.

[13] Chongkai Gao, Haozhuo Zhang, Zhixuan Xu, Zhehao Cai, and Lin Shao. Flip: Flow-centric generative planning for general-purpose manipulation tasks. *arXiv preprint arXiv:2412.08261*, 2024.

[14] Haoran Geng, Feishi Wang, Songlin Wei, Yuyang Li, Bangjun Wang, Boshi An, Charlie Tianyue Cheng, Haozhe Lou, Peihao Li, Yen-Jen Wang, Yutong Liang, Dylan Goetting, Chaoyi Xu, Haozhe Chen, Yuxi Qian, Yiran Geng, Jiageng Mao, Weikang Wan, Mingtong Zhang, Jiangran Lyu, Siheng Zhao, Jiazhao Zhang, Jialiang Zhang, Chengyang Zhao, Haoran Lu, Yufei Ding, Ran Gong, Yuran Wang, Yuxuan Kuang, Ruihai Wu, Baoxiong Jia, Carlo Sferrazza, Hao Dong, Siyuan Huang, Yue Wang, Jitendra Malik, and Pieter Abbeel. Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning, 2025.

[15] Théophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: 3d feature field transformers for multi-task robotic manipulation. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *CoRL*, Proceedings of Machine Learning Research. PMLR, 2023.

[16] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, and Yashraj S. Narang. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *ICRA*. IEEE, 2023.

[17] Tairan He, Zhengyi Luo, Xialin He, Wenli Xiao, Chong Zhang, Weinan Zhang, Kris M. Kitani, Changliu Liu, and Guanya Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. In *CoRL*, 2024.

[18] Xialin He, Runpei Dong, Zixuan Chen, and Saurabh Gupta. Learning getting-up policies for real-world humanoid robots. In *Robotics: Science and Systems XXI, Los Angeles, California, June 21-25, 2025*, 2025.

[19] Cheng-Chun Hsu, Zhenyu Jiang, and Yuke Zhu. Ditto in the house: Building articulation models of indoor scenes through interactive perception. In *ICRA*, 2023.

[20] Jiahui Huang, Zan Gojcic, Matan Atzmon, Or Litany, Sanja Fidler, and Francis Williams. Neural kernel surface reconstruction. In *CVPR*, 2023.

[21] Vidhi Jain, Maria Attarian, Nikhil J. Joshi, Ayzaan Wahid, Danny Driess, Quan Vuong, Pannag R. Sanketi, Pierre Sermanet, Stefan Welker, Christine Chan, Igor Gilitschenski, Yonatan Bisk, and Debidatta Dwibedi. Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers. In *Robotics: Science and Systems (RSS)*, 2024. URL `https://roboticsconference.org/2024/program/papers/52/`.

[22] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *CVPR*. IEEE, 2022.

[23] Tsung-Wei Ke, Nikolaos Gkanatsios, and Katerina Fragkiadaki. 3d diffuser actor: Policy diffusion with 3d scene representations. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard, editors, *CoRL*, Proceedings of Machine Learning Research. PMLR, 2024.

[24] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, (4), 2023.

[25] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. RMA: rapid motor adaptation for legged robots. In *Robotics: Science and Systems XVII, Virtual Event, July 12-16, 2021*, 2021.

[26] Hang Lai, Weinan Zhang, Xialin He, Chen Yu, Zheng Tian, Yong Yu, and Jun Wang. Sim-to-real transfer for quadrupedal locomotion via terrain transformer. In *ICRA*. IEEE, 2023.

[27] Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Sci. Robotics*, (26), 2019.

[28] Jiahui Lei, Yijia Weng, Adam Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. *arXiv preprint arXiv:2405.17421*, 2024.

[29] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *ECCV*. Springer, 2024.

[30] Tianye Li, Mira Slavcheva, Michael Zollhöfer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard A. Newcombe, and Zhaoyang Lv. Neural 3d video synthesis from multi-view video. In *CVPR*. IEEE, 2022.

[31] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. Megasam: Accurate, fast and robust structure and motion from casual dynamic videos. In *arxiv*, 2024.

[32] Zhongyu Li, Xue Bin Peng, Pieter Abbeel, Sergey Levine, Glen Berseth, and Koushil Sreenath. Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control. *The International Journal of Robotics Research*, 2024.

[33] William Liang, Sam Wang, Hung-Ju Wang, Osbert Bastani, Dinesh Jayaraman, and Yecheng Jason Ma. Eurekaverse: Environment curriculum generation via large language models. *arXiv preprint arXiv:2411.01775*, 2024.

[34] Matthew Loper and Michael J. Black. Opendr: An approximate differentiable renderer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 154–169. Springer, 2014. doi: 10.1007/978-3-319-10584-$0_1$1.

[35] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj S. Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. In *CoRL*, Proceedings of Machine Learning Research. PMLR, 2023.

[36] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. 2022.

[37] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, (1), 2021.

[38] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, (1), 2022.

[39] Nikhil Mishra, Maximilian Sieb, Pieter Abbeel, and Xi Chen. Closing the visual sim-to-real gap with object-composable nerfs. In *ICRA*. IEEE, 2024.

[40] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *ICCV*, 2019.

[41] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *ICRA*. IEEE, 2018.

[42] Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. Unidepthv2: Universal monocular metric depth estimation made simpler. *arXiv preprint arXiv:2502.20110*, 2025.

[43] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. In Hadas Kress-Gazit, Siddhartha S. Srinivasa, Tom Howard, and Nikolay Atanasov, editors, *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018.

[44] Cédric Portaneri, Mael Rouxel-Labbé, Michael Hemmer, David Cohen-Steiner, and Pierre Alliez. Alpha Wrapping with an Offset. *ACM Transactions on Graphics*, (4), June 2022.

[45] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-hand object rotation via rapid motor adaptation. In *CoRL*, Proceedings of Machine Learning Research. PMLR, 2022.

[46] Mohammad Nomaan Qureshi, Sparsh Garg, Francisco Yandun, David Held, George Kantor, and Abhishesh Silwal. Splatsim: Zero-shot sim2real transfer of rgb manipulation policies using gaussian splatting, 2024.

[47] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763, 2021.

[48] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024.

[49] Tianhe Ren, Yihao Chen, Qing Jiang, Zhaoyang Zeng, Yuda Xiong, Wenlong Liu, Zhengyu Ma, Junyi Shen, Yuan Gao, Xiaoke Jiang, Xingyu Chen, Zhuheng Song, Yuhong Zhang, Hongjie Huang, Han Gao, Shilong Liu, Hao Zhang, Feng Li, Kent Yu, and Lei Zhang. Dino-x: A unified vision model for open-world object detection and understanding, 2024.

[50] Jun Shi, Yong A, Yixiang Jin, Dingzhe Li, Haoyu Niu, Zhezhu Jin, and He Wang. Asgrasp: Generalizable transparent object reconstruction and 6-dof grasp detection from RGB-D active stereo camera. In *ICRA*. IEEE, 2024.

[51] Jonah Siekmann, Yesh Godse, Alan Fern, and Jonathan W. Hurst. Sim-to-real learning of all common bipedal gaits via periodic reward composition. In *ICRA*. IEEE, 2021.

[52] Jonah Siekmann, Yesh Godse, Alan Fern, and Jonathan W. Hurst. Sim-to-real learning of all common bipedal gaits via periodic reward composition. In *ICRA*. IEEE, 2021.

[53] Balakumar Sundaralingam, Siva Kumar Sastry Hari, Adam Fishman, Caelan Garrett, Karl Van Wyk, Valts Blukis, Alexander Millane, Helen Oleynikova, Ankur Handa, Fabio Ramos, Nathan Ratliff, and Dieter Fox. curobo: Parallelized collision-free minimum-jerk robot motion generation, 2023.

[54] Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, et al. Neuralfeels with neural fields: Visuotactile perception for in-hand manipulation. *Science Robotics*, (96), 2024.

[55] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. In *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018.

[56] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*. IEEE, 2017.

[57] Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation. *arXiv preprint arXiv:2403.03949*, 2024.

[58] Alexander Veicht, Paul-Edouard Sarlin, Philipp Lindenberger, and Marc Pollefeys. GeoCalib: Single-image Calibration with Geometric Optimization. In *ECCV*, 2024.

[59] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024.

[60] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *CVPR*, 2025.

[61] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024.

[62] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. doi: 10.1109/TIP.2003.819861.

[63] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. FoundationPose: Unified 6d pose estimation and tracking of novel objects. In *CVPR*, 2024.

[64] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*, 2024.

[65] Yuxuan Wu, Lei Pan, Wenhua Wu, Guangming Wang, Yanzi Miao, Fan Xu, and Hesheng Wang. Rl-gsbridge: 3d gaussian splatting based real2sim2real method for robotic manipulation learning. *arXiv preprint arXiv:2409.20291*, 2024.

[66] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. *arXiv preprint arXiv:2412.01506*, 2024.

[67] Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard, editors, *CoRL*, Proceedings of Machine Learning Research. PMLR, 2024.

[68] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024.

[69] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *NeurIPS*, 2024.

[70] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *CVPR*, 2024.

[71] Weirui Ye, Fangchen Liu, Zheng Ding, Yang Gao, Oleh Rybkin, and Pieter Abbeel. Video2policy: Scaling up manipulation tasks in simulation through internet videos, 2025. URL `https://arxiv.org/abs/2502.09886`.

[72] Yanjie Ze, Ge Yan, Yueh-Hua Wu, Annabella Macaluso, Yuying Ge, Jianglong Ye, Nicklas Hansen, Li Erran Li, and Xiaolong Wang. Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In Jie Tan, Marc Toussaint, and Kourosh Darvish, editors, *CoRL*, Proceedings of Machine Learning Research. PMLR, 2023.

[73] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arxiv:2410.03825*, 2024.

Supplementary Material

Contents

A Implementation Details

In this section, we introduce the implementation details of ROSE. To be specific, we present the details of the Real2Sim transfer of human videos, Sim2Real policy training, and real-world setup in Appendix A.1, Appendix A.2, and Appendix A.3, respectively.

A.1 Real-to-Sim Transfer

We use the videos with a resolution of 512×288 containing 60$\sim$300 frames in 30 FPS (spanning 2s to 10s). We first reconstruct the 3D point clouds by running MegaSaM [31], detailed as follows.

Point Cloud Reconstruction For every frame, we follow MegaSaM [31] to get the affine-invariant monocular disparity map with Depth-Anything V2 [68], a camera pose in the world coordinate, and the focal length estimation obtained with UniDepth V2 [42]. With this information, we align the disparity to the metric scale, which is converted to the pixel-aligned 3D point clouds coordinated in the world frame. Afterwards, we use GeoCalib [58] on the first frame to estimate the scene's gravity direction. We then rotate the camera rig so that the estimated gravity is to the negative of the z-axis in a right-hand coordinate system, resulting in gravity-aligned point clouds in the world frame. To further avoid residual artifacts, we apply edge dilation to remove colors at boundaries and depth-gradient pruning to remove point clouds with corresponding gravity exceeding a certain threshold of 0.8 for reducing depth discontinuities.

For every image, we use Ground-SAM-2 [48] with DINO-X-Track[49] to obtain binary masks of objects in the scene, used for object removal and reconstruction. For the scene point cloud, we fuse the point clouds by random sampling over the video to leverage complementary information to reduce the inaccurate point clouds caused by occlusions and partial scenes after object removal. The random sampling is used for balancing every frame's contribution. On average, the point cloud reconstruction would take 3 minutes to process a 5-second casual video.

3D Mesh Reconstruction For the object mesh, we use TRELLIS [66] to reconstruct the 3D mesh, and we set the α channel's threshold to 0.2 to help reconstruct dark objects. For the scene mesh, we use a three-stage method. In the first stage, we run Neural Kernel Surface Reconstruction (NKSR) [20] for surface fitting. To close the small gaps created by mask removal while preserving high-frequency geometry, we set the detail level to 0.4 and use a single MISE iteration. Afterwards, to make the mesh orientable, two-manifold, and self-intersection-free for simulator usage, we wrap the resulting mesh with CGAL alpha-wrapping algorithm [44]. The α value is set to 400. Finally, each mesh vertex v of $\mathcal{M}_{\text{wrap}}$ inherits the RGB value $c(\cdot)$:

$$c(v) = \sum_{p \in \mathcal{N}_k(v)} w(p,v)\, c(p), \quad w(p,v) \propto \frac{1}{\|p - v\|_2}.$$

This is the inverse-distance-weighted average of its k-nearest neighbours $\mathcal{N}_k(v)$ in the point cloud. In this work, we use $k = 3$ with a maximum distance of 5cm.

Improved Foundation Pose Foundation pose [63] requires that the scale of the mesh and the scale, unprotected from the depth map, be the same in order to ensure accurate pose estimation. Therefore, we align the trellis mesh $\mathbf{M}^{\text{trellis}}$ with the scene mesh $\mathbf{M}^{\text{scene}}$ by the following transformation:

$$\mathbf{M}^{\text{trellis-align}} = \mathbf{G}_0 \mathcal{Q} \mathbf{s} \mathbf{M}^{\text{trellis}} f, \tag{6}$$

where s is the scale factor, $\mathbf{G}_0$ is the camera pose, $\mathcal{Q}$ is the object pose, and f is the focal length.

For pose tracking, we begin by using the scale $\mathbf{s}$ estimated in the first frame to adjust the scale of $\mathbf{M}^{\text{trellis}}$. We then follow the approach outlined in FoundationPose [63]. The object pose $\mathcal{Q}_i$ is initialized using the previously estimated pose $\mathcal{Q}_{i-1}$, and the refinement network is applied to further refine $\mathcal{Q}_i$, yielding the final estimation of the object pose for the current frame.

A.2 Sim-to-Real Policy

Simulation Environment Setup We use RoboVerse [14] as our simulation platform to establish the data collection pipeline. Specifically, we adopt the IsaacGym branch for mesh loading, policy execution, and reinforcement learning, while leveraging the IsaacLab branch for high-fidelity rendering and vision-based policy training.

Simulation and control parameters follow the default settings provided by RoboVerse. For both objects and scenes, we set the friction coefficient to 0.5. To ensure accurate collision detection and reliable physics simulation, we apply convex decomposition to the scene geometry.

Robotic Data Collection based on Motion Planning We utilize GSNet [40] and cuRobo [53] as the core components of our motion planning pipeline, and conduct all simulations within the Isaac Gym environment. After reconstructing the scene (as described in Appendix A.1), we load the reconstructed environment and the target object mesh into the simulator. The object is represented using a high-resolution mesh, preserving geometric detail necessary for accurate grasp prediction and motion planning.

Using GSNet, we extract both the surface point cloud and a predicted grasp pose for the target object. These predictions are then used to initialize an inverse kinematics (IK) solver provided by cuRobo, which computes a feasible trajectory for the robot's end effector to reach the designated grasping configuration. During this process, we account for kinematic constraints, joint limits, and potential collisions in the environment.

Upon reaching the grasp pose, the robot executes a grasp based on a set of hand-crafted heuristics, which evaluate grasp stability using factors such as contact normals and finger placement. Once the grasp is completed, the robot follows a trajectory generated by a previously introduced motion prediction model, which guides the object to a specified goal position or task-specific location.

Robotic Data Collection based on Reinforcement Learning Our method adopts a two-stage policy for robotic manipulation. In the first stage, we employ a reinforcement learning (RL) approach to train a policy that guides the robot's end effector toward the object and performs a grasp once proximity is sufficiently close. To facilitate generalization across different grippers or robotic hands, we design a simple yet broadly applicable reward function. This reward encourages the end effector to reduce its distance to the target object and penalizes undesired motions, without relying on gripper-specific parameters, making it adaptable to a wide range of hardware configurations.

During deployment, we execute the learned grasping policy for a fixed horizon of 50 steps, under the assumption that a successful grasp is achieved by the end of this phase. In the second stage, we switch to a motion planning phase using cuRobo [53]. The robot follows a precomputed trajectory that guides the grasped object to its goal location or completes the assigned task. This two-stage setup decouples grasp acquisition from subsequent manipulation, allowing each component to be optimized independently while ensuring end-to-end effectiveness.

A.3 Real-World Experimental Setups

Franka Setting For our real-world experiments, we use a Franka Emika Panda robotic arm equipped with a Robotiq 2F-85 adaptive gripper. This setup provides a reliable and widely used platform for evaluating grasping and manipulation policies in physical environments. The robot is controlled via a high-level interface that integrates seamlessly with our planning and control pipeline.

To reconstruct the environment, we capture RGB-D data using both an iPhone 16 Pro and a DJI Osmo Pocket 3. These consumer-grade devices offer high-resolution color and depth sensing capabilities, allowing for efficient and accessible scene scanning.

B Real-to-sim Benchmark

B.1 benchmark evaluation metric details

Our proposed benchmark is based on three primary evaluations: scene reconstruction similarity, object reconstruction similarity, and object trajectory reconstruction.

Uniform Sampling. Uniform sampling extracts a point cloud from a mesh by taking N points drawn *i.i.d.* over the surface of the mesh $\mathbf{M}$. Unless stated otherwise, $N = 10{,}000$ in all our experiments.

Symmetric Chamfer Distance. Given two point clouds A and B, symmetric chamfer distance is defined as:

$$\mathrm{ChamferDist}(A, B) = \frac{1}{|A|}\sum_{a\in A}\min_{b\in B}\|a-b\|_2^2 \;+\; \frac{1}{|B|}\sum_{b\in B}\min_{a\in A}\|b-a\|_2^2.$$

Scene Reconstruction Similarity. To evaluate scene reconstruction similarity, we first construct a point cloud $\mathbf{P}^{\text{scene}}$ through uniform sampling the reconstructed mesh. We then align $\mathbf{P}^{\text{scene}}$ with each frame's ground-truth point cloud $\mathbf{P}_i^{\text{gt-scene}}(i = 1, \ldots, F)$ by optimizing an SE(3) transformation to yield $\hat{\mathbf{P}}^{\text{scene}}$. Finally, we compute the average chamfer distance across the aligned frame point cloud $\hat{\mathbf{P}}^{\text{scene}}$ and the ground-truth scene point clouds.

$$\mathbf{E}_{\text{scene}} = \frac{1}{F}\sum_{i=1}^{F}\text{ChamferDist}(\mathbf{P}_i^{\text{gt-scene}}, \hat{\mathbf{P}}^{\text{scene}})$$

Object Reconstruction Similarity. Similarly to scene reconstruction similarity, we construct point clouds $\mathbf{P}^{\text{gt-obj}}$ and $\mathbf{P}^{\text{obj}}$ by uniformly sampling the ground-truth and reconstructed object meshes respectively. We then align the point clouds by optimizing an SE(3) transformation to yield $\hat{\mathbf{P}}^{\text{obj}}$. Finally, we evaluate object reconstruction similarity as the chamfer distance between the aligned point clouds.

$$\mathbf{E}_{\text{obj}} = \text{ChamferDist}(\mathbf{P}^{\text{gt-obj}}, \hat{\mathbf{P}}^{\text{obj}})$$

Trajectory Reconstruction. We evaluate object trajectory reconstruction using three metrics: absolute pose error translation (APE$_{trans}$), relative pose error translation (RPE$_{trans}$) and relative pose error rotation (RPE$_{rot}$). We compute these between the ground truth trajectory $\mathbf{Q}^{\text{gt-obj}}$ and the reconstructed trajectory $\mathbf{Q}^{\text{obj}}$, after aligning the reconstructed trajectories scale and SE(3) transformations.

The absolute pose error (APE) measures the deviation between corresponding poses and is defined as:

$$e_{\text{ape}}(\mathbf{Q}^{\text{gt-obj}}, \mathbf{Q}^{\text{obj}}) = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathbf{Q}_{xyz}^{\text{gt-obj}} - \hat{\mathbf{Q}}_{xyz}^{\text{obj}}\right\|_2$$

where $\mathbf{Q}_{xyz}$ denotes the translation component of pose $\mathbf{Q}_i$ and $\hat{\mathbf{Q}}$ denotes the aligned trajectory.

The relative pose error (RPE) measures the local consistency of motion between consecutive poses. For an interval $\Delta = 1$, we define the relative transformations as:

$$\mathbf{T}_i^{\text{gt-rel}} = (\mathbf{Q}_i^{\text{gt-obj}})^{-1}\mathbf{Q}_{i+\Delta}^{\text{gt-obj}}$$
$$\mathbf{T}_i^{\text{obj-rel}} = (\hat{\mathbf{Q}}_i^{\text{obj}})^{-1}\hat{\mathbf{Q}}_{i+\Delta}^{\text{obj}}$$

The translation component of the RPE at time i is given by:

$$e_{\text{rpe,trans}}(i) = \left\|\text{trans}\left((\mathbf{T}_i^{\text{gt-rel}})^{-1}\mathbf{T}_i^{\text{obj-rel}}\right)\right\|_2$$

where $\text{trans}(\cdot)$ extracts the translation part of a transformation.

The rotation component of the RPE is given by:

$$e_{\text{rpe,rot}}(i) = \arccos\left(\frac{\text{trace}\left(\text{rot}\left((\mathbf{T}_i^{\text{gt-rel}})^{-1}\mathbf{T}_i^{\text{obj-rel}}\right)\right) - 1}{2}\right)$$

where $\text{rot}(\cdot)$ extracts the rotation matrix component of a transformation.

B.2 Data Details

We select 5 representative tasks from CALVIN [36] implemented in RoboVerse [14], covering the basic manipulation tasks on rigid objects: lift, push, rotate, "pick and place", and unstack blocks. These tasks are performed on top of a delicate desk, allowing for evaluating the proposed pipeline by reconstructing both the scene (desk) and the objects in interest (blocks).

B.3 Qualitative Results

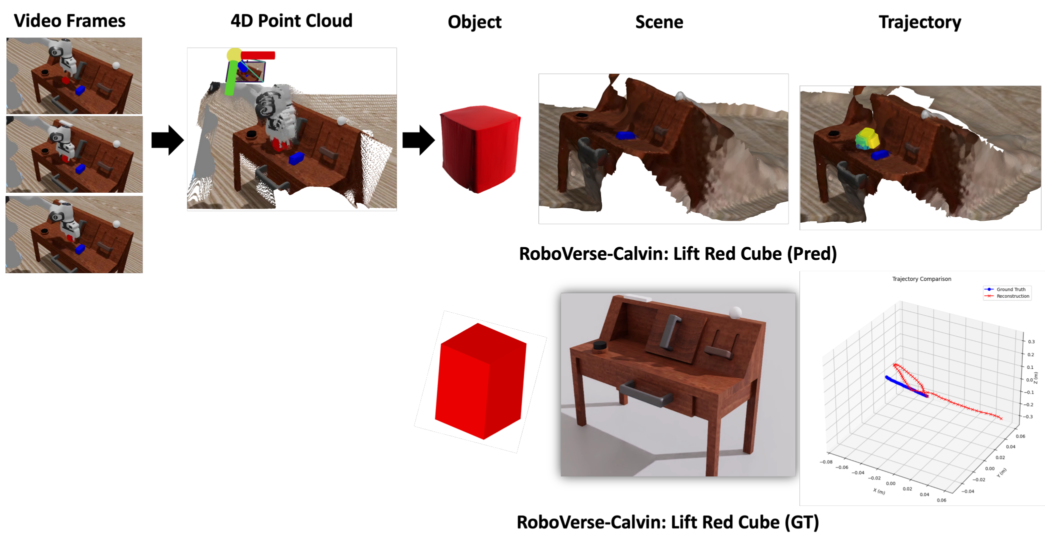

Figure 4: **Qualitative examples of our real-to-sim benchmark.**