Test-Time Spatial Reasoning for Robot Manipulation Using Generative Real-to-Sim

Ivan Kapelyukh[1,2], Yafei Hu[1], Ran Gong[1], Brandon May[1], Tushar Kusnur[1], Laura Herlant[1], Karl Schmeckpeper[1], Edward Johns[2,§], Xiaohan Zhang[1,§]
[1]Robotics and AI Institute [2]Imperial College London [§]Project Leads

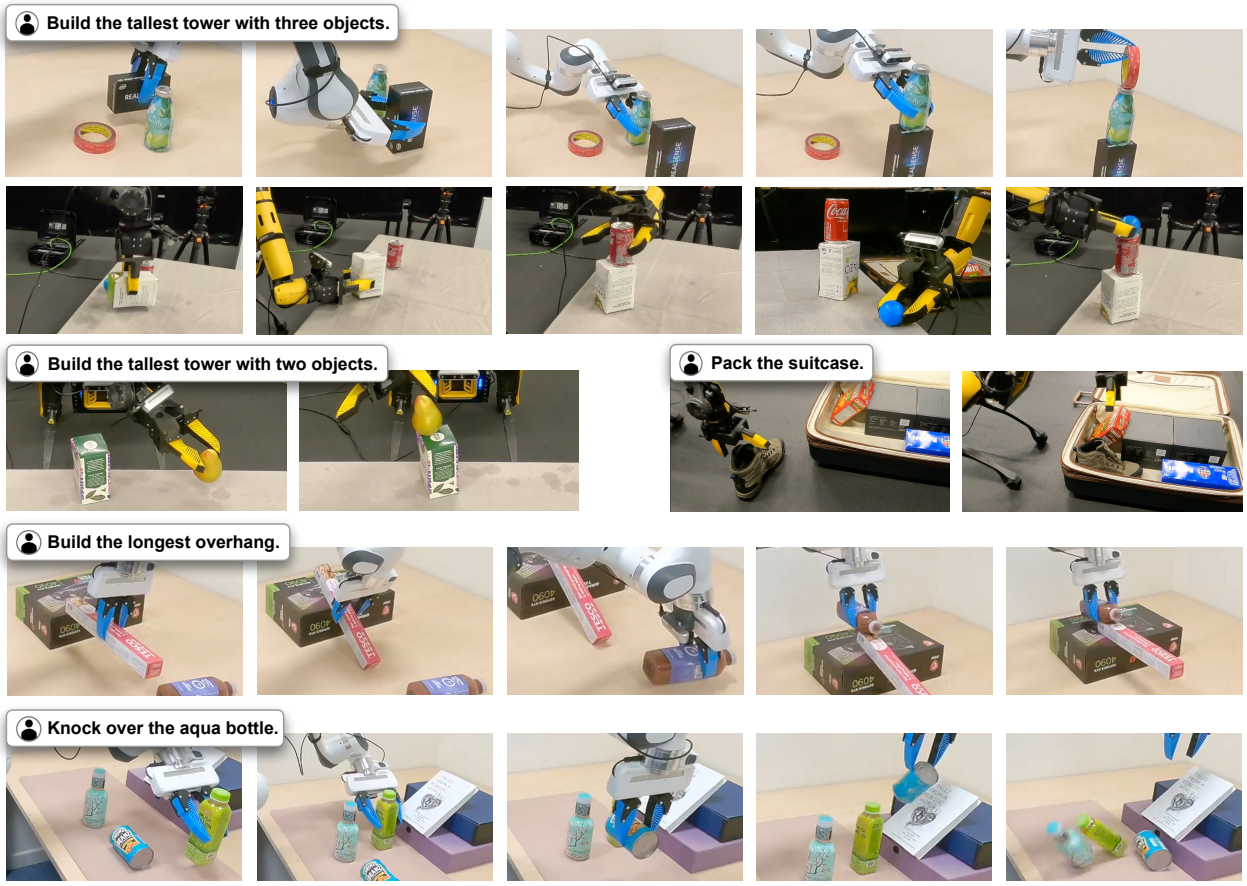

Fig. 1. Zero-shot spatial and physical reasoning with Simify. From a single RGB-D view and task reward, the robot reconstructs the scene, searches for object arrangements in simulation, and then executes the rearrangement in the real world.

***Abstract*— Spatial reasoning is fundamental to general robot intelligence, as it enables robots to complete long-horizon tasks involving multi-object interaction. We introduce Simify, a training-free, test-time framework that performs explicit spatial reasoning via massively parallel physics simulation. From a single RGB-D image of a scene, Simify reconstructs simulation-ready assets leveraging 3D generative models and vision-language models. Then given a task specified by a reward function (e.g., *build the tallest tower*), Simify launches thousands of parallel rollouts in simulation and performs an evolutionary search to optimize object arrangements, typically converging within seconds. We conduct quantitative experiments on real-robot hardware to demonstrate the ability of our framework to execute complex object rearrangement tasks end-to-end with previously unseen objects. Results show that our framework outperforms prior work on foundation models for spatial reasoning by effectively exploiting large-scale parallel simulation during inference, and also highlight the importance of complete and accurate geometry for successful sim-to-real transfer.**

I. Introduction

Spatial reasoning refers to the ability to understand object geometry, spatial relationships, and feasible arrangements in 3D space in order to achieve a goal. For robots, spatial reasoning must often be physically grounded: the robot must not only infer where objects can be placed, but also whether the resulting arrangement will remain stable, produce the desired contact interactions, and be executable in the real world. Despite rapid progress in training spatial-aware Vision-Language Models (VLMs), replicating such physically grounded spatial reasoning in continuous 3D space remains a fundamental challenge for robots.

To endow robots with spatial reasoning capability, recent works have largely explored two directions: using pre-trained 3D-aware foundation models ([1], [2], [3], [4], [5]), or training end-to-end Vision-Language-Action (VLA) policies ([6],

[7], [8]). However, geometric accuracy and physical grounding remain major bottlenecks for these approaches: while they capture high-level semantic relationships, tasks like stacking and balancing (Fig. 1) require high precision and strict adherence to physical dynamics – capabilities that implicit learned representations often fail to guarantee. On the other hand, VLAs and other imitation learning approaches are developed by scaling data to solve a diverse set of such manipulation tasks. Despite the large amount of teleoperation data they require during training, these approaches tend not to discover new strategies on novel tasks at test time; instead, their generalization is mostly imitation of action patterns similar to those the robot has seen before [9].

In this work, we focus on a complementary regime to policy learning: test-time spatial reasoning for novel object rearrangement. Rather than assuming task-specific demonstrations, offline training data of successful arrangements, known object models, or a learned goal-state predictor, we ask whether a robot can construct a physical model of the current scene and search for a solution directly at test time. This setting is especially important for object and position generalization: when previously unseen objects or new initial poses change the feasible placement order, orientation, or contact configuration, success may require reasoning about the specific geometry and physical interactions in the current scene. While imitation learning and RL are powerful when sufficient task distributions and training data are available, they are not the focus of this paper. Instead, our method is closer in spirit to test-time planning and TAMP-style reasoning, but targets open-world scenes where object models must be inferred from a single RGB-D observation.

As humans, we approach such problems through a process of deliberation: we can mentally rotate, place, and evaluate candidates before taking any actions, even when encountering the objects for the first time. Analogously, robots should also simulate and search for solutions at test time and plan accordingly. This paradigm shifts the burden from learning all geometry and physics inside one visuomotor policy to using external tools: *a 3D physics simulator and a search-based optimization procedure for explicit reasoning.*

We introduce Simify, a training-free framework for solving tasks that require spatial creativity (Fig. 2). From a single RGB-D image, Simify reconstructs a simulation-ready digital twin of previously unseen environments using 3D generative priors and LLM/VLMs. Specifically, we leverage open-vocabulary segmentation and 2D inpainting models to resolve occlusions, feeding the resulting object images into an image-to-3D diffusion model to generate complete, scaled meshes. Once instantiated in a massively parallel physics simulator, Simify uses an evolutionary algorithm to jointly optimize the discrete placement sequence and continuous goal poses for each object. By evaluating thousands of noisy physical rollouts in parallel, the framework effectively searches for robust, executable arrangements that maximize the reward function given for the task. Finally, the optimal arrangement is transferred back to the real world and executed via a vision-based, collision-free motion planning pipeline.

We quantitatively evaluate Simify both in simulation and on real-robot hardware. Results show the importance of complete and accurate geometry for downstream task success, and highlight how our framework can be applied to a range of spatial reasoning tasks.

II. Related work

A. Generative Real-to-Sim

Real-to-sim methods build a simulation of the scene observed by the robot, enabling the robot to reason, learn, or plan in a reconstructed physical environment before acting in the real world. Prior work has studied real-to-sim for estimating physical properties such as mass, friction, contact dynamics, and articulation [10], [11], [12], [13], [14], as well as visual reconstruction using depth scans, NeRFs, or Gaussian Splatting [15], [16], [17]. However, these approaches often require known object models, dense multi-view scanning, manual setup, or multiple cameras, and they struggle to recover occluded object geometry from a single robot observation. In this work, we use recent image-to-3D generative models as part of our real-to-sim pipeline: given a single RGB-D view, the robot segments objects, completes occluded regions, generates full object meshes with a visual generative prior, scales and registers them to the observed scene, and instantiates them in physics simulation. By leveraging powerful vision priors, our framework minimizes human effort and enables the reconstruction of unobserved surfaces (e.g. the bottom of an object), which we show is important for accurate dynamics in simulation.

B. Real-to-Sim in Robotics

Several recent robotics systems have also begun to use image-to-3D models or real-to-sim techniques for manipulation, including grasp sampling, reconstruction of human demonstrations, policy evaluation, dataset generation, and rapid reinforcement learning in simulation [18], [19], [20], [21], [22], [23], [24]. These works are complementary to ours, but they primarily use the generated simulation to train or evaluate policies, often assuming that the task goal or demonstration is already provided. For example, ZeroBot [24] uses generative real-to-sim to train a state-based RL policy that manipulates a novel object to a given goal pose. In contrast, our work studies a different regime: test-time spatial reasoning, where the robot must infer the goal arrangement itself. To our knowledge, this is the first work to explore single-view generative real-to-sim for test-time physics-based planning to solve these long-horizon, multi-object spatial reasoning tasks, with no human demonstrations or training datasets of arrangements required.

C. Object Rearrangement and TAMP

Object rearrangement requires choosing both a discrete placement order and continuous goal poses, making it a natural setting for spatial reasoning [25]. Task and Motion Planning (TAMP) is a strong framework for such long-horizon manipulation problems, as it jointly reasons over

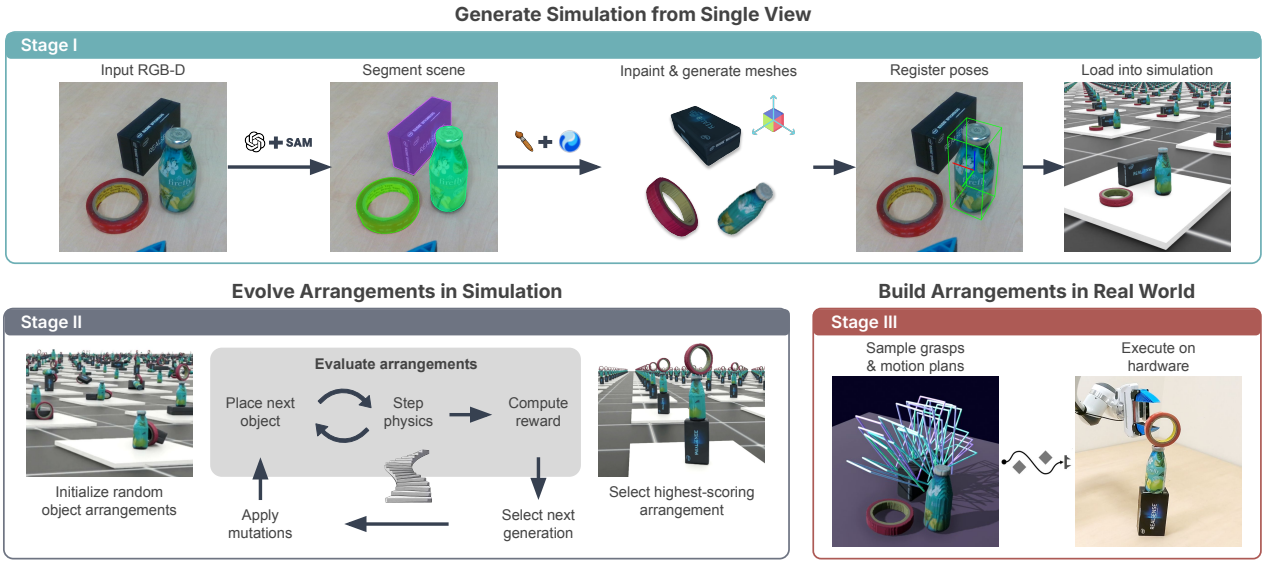

Fig. 2. Overview of Simify. (1) From a single RGB-D view, the system segments and completes objects, generates scaled meshes, estimates their poses, and builds a parallel physics simulation. (2) Evolutionary search jointly optimizes placement order and goal object poses, followed by noisy validation to select a robust, high-reward arrangement. (3) The robot tracks objects, samples grasps, and executes collision-free pick-and-place motions.

symbolic action sequences and continuous geometric feasibility [26], [27]. Recent open-vocabulary TAMP systems further use VLMs or LLMs to ground language instructions, infer goals, or propose constraints for planning [28], [29]. We therefore compare against LLM-GROP as a strong zero-shot TAMP baseline. Our setting differs from most prior rearrangement work in that success depends not only on geometric feasibility, but also on physical outcomes after execution, such as stability, balance, and dynamic object-object interaction. Traditional TAMP methods often assume known or reconstructed object models and hand-designed predicates or constraints, while learned rearrangement methods require training datasets of successful arrangements, thus requiring more human effort to apply to a new task [30], [31], [27]. In contrast, our method is training-free: from a single RGB-D observation and a task reward, it reconstructs the current scene and uses massively parallel physics simulation at test time to search over placement order and continuous object poses. This enables real-to-sim-to-real spatial reasoning with everyday objects, including tasks such as *Stacking*, *Cantilever*, and *Dominoes*.

III. Problem Statement

We consider the challenging multi-object rearrangement setup, and we specifically tackle this problem in environments containing novel, everyday objects without relying on human demonstrations or object models known a priori.

Sensory Input & Task Specification: The robot is provided with an initial state of the environment, captured by a single-view RGB-D observation, denoted as I. The task specification $\mathcal{T}$ is provided either as a natural-language description or a reward function. In this work, we formalize the task as a scalar reward function $R_{\mathcal{T}}$[1] that evaluates the geometric and structural success of the post-execution scene.

Action & Solution: Solving the rearrangement task requires the robot to output a single ordered sequence of N desired object goal poses, denoted as $\mathbf{G} = (G_1, G_2, \ldots, G_N)$. The order of this sequence intrinsically defines the discrete placement sequence, while each G_i specifies the continuous 6D target pose for that object i in the world frame.

We assume the robot sequentially executes these goal poses using a vision-based pick-and-place skill. While this skill could be an end-to-end visuomotor policy, in this work we implement it via a traditional pipeline integrating pose estimation, grasp sampling, and collision-free motion planning. More details can be found in Section IV.

Dynamics & Objective: Real-world physical execution is inherently non-deterministic. After a robot places an object at its goal pose G_i, it settles into a final pose $\tilde{G}_i$ due to physical interactions, ranging from quasi-static structural settling to dynamic multi-object collisions (e.g., *Dominoes*). We denote this "place-then-settle" physical transition as a mapping $\mathcal{S}$, which induces the final settled poses based on the ordered sequence of object goal poses: $\tilde{G} = \mathcal{S}(\mathbf{G})$.

The overarching objective is to discover an executable rearrangement sequence $\mathbf{G}^*$ that maximizes the expected task reward on the fully settled scene:

$$\mathbf{G}^* = \arg\max_{\mathbf{G}} \mathbb{E}\Big[R_{\mathcal{T}}(\tilde{\mathbf{G}})\Big] \quad \text{s.t.} \quad \tilde{\mathbf{G}} = \mathcal{S}(\mathbf{G}).$$

We rely on simulated physics to estimate $\mathcal{S}$ prior to real-world execution. In this work, we focus on real2sim for geometry, and assume that physics parameters such as mass and friction are given, following prior work [15].

IV. Methodology

In order to address this problem, we propose the Simify framework, an overview of which is shown in Fig. 2. First, our method builds a simulation of the scene using mesh

[1]See Section V for reward function definitions. We assume $R_{\mathcal{T}}$ is given; synthesizing rewards from language is left for future work [32], [33].

generation (Section IV-A). Once in simulation, the poses of the objects are optimized using an evolutionary algorithm, in order to find an arrangement with a high task reward (Section IV-B). After the goal pose for each object has been determined, the rearrangement is executed on the real robot using motion planning (Section IV-C).

A. Real-to-sim

The objective of this stage is to construct a simulation-ready mesh for each object. The robot captures a single-view RGB-D image I using a wrist-mounted camera; we assume a given viewpoint and leave autonomous active vision to future work [34]. To reconstruct the scene without task-specific instructions or predefined object names, we use bottom-up segmentation. SAM-2 [35] first generates segment proposals, which may over-segment objects or include background regions. A VLM (GPT-4o in our experiments) lists the manipulable objects in I and assigns the numbered SAM-2 segments to each object. This produces manipulation-relevant object masks with minimal human input, while allowing optional user instructions to modify the segmentation.

Because image-to-3D models such as Hunyuan3D-2 [36] are typically trained on unoccluded object views, we first complete each masked object crop using a 2D amodal inpainting model [37] based on Stable Diffusion [38]. The model receives the object crop with other segments masked out and a VLM-generated text description, producing an occlusion-free image for mesh generation. Since the generated mesh lacks metric scale, we align it to the observed partial depth point cloud using ICP. We then register the scaled mesh to the scene with Foundation Pose, which estimates the initial object pose T_{WO} from the RGB-D image and generated mesh. The resulting meshes and poses are loaded into Isaac Lab [39] for massively parallel GPU-accelerated simulation. Our pipeline also supports multiple views by tracking segments with SAM-2 and aggregating depth point clouds for scale estimation; however, we evaluate the single-view setting because it avoids additional scanning and matches the input regime of current image-to-3D models.

B. Evolutionary Optimization in Simulation

Objective. Given reconstructed object meshes and a task reward, we optimize the placement order and 6D goal poses to maximize the final arrangement reward. We use an evolutionary algorithm (Algorithm 1) because the search space is mixed discrete–continuous, multimodal, and well suited to massively parallel simulation.

Evolution. We initialize L solutions with random placement orders and object poses (Fig. 2). Each solution is evaluated in a parallel simulation by sequentially placing the objects, allowing the scene to settle after each placement, and computing the final task reward. The top 5% are retained as elites, while the remaining solutions are sampled according to a reward-based softmax and mutated in both pose and placement order. This process favors high-reward solutions while maintaining population diversity.

Algorithm 1: Evolutionary Optimization of Multi-Object Rearrangement

Input: N object (meshes) $\{\mathbf{M}^{(1)}, \dots, \mathbf{M}^{(N)}\}$, Reward function $R_{\mathcal{T}}$, Population size L, Max generations γ, Validation rollouts K
Output: $\mathbf{G}^*$
Initialize population $\mathcal{P}_0 = \{\mathbf{G}^{(1)}, \dots, \mathbf{G}^{(L)}\}$;
for $g \leftarrow 0$ **to** $\gamma - 1$ **do**
  foreach $\mathbf{G} \in \mathcal{P}_g$ ***in parallel*** **do**
    Reset simulation;
    for *object* $i \leftarrow 1$ **to** N **do**
      Add Gaussian noise: $G_i = G_i + \epsilon$;
      Place object i at G_i;
      Step physics until stable;
    Compute task reward r;
  Rank population $\mathcal{P}_g$ by r;
  Keep top 5% of $\mathbf{G} \in \mathcal{P}_g$ unchanged for $\mathcal{P}_{g+1}$;
  Sample the remaining via softmax over rewards;
  Mutate goal poses and placement order;
Validation Stage:;
foreach $\mathbf{G} \in \mathcal{P}_\gamma$ ***in parallel*** **do**
  for $k \leftarrow 1$ **to** K **do**
    Execute $\mathbf{G}$ with ϵ;
    Obtain $\tilde{G}$ and compute reward $r_k = R_{\mathcal{T}}(\tilde{G})$;
  Compute expected reward: $\bar{r} = \frac{1}{K}\sum_{k=1}^{K} r_k$;
return $\mathbf{G}^* = \arg\max_{\mathbf{G}} \bar{r}$;

Validation. To account for reconstruction and execution errors, Gaussian placement noise is applied during optimization. After the final generation, each candidate is evaluated over multiple noisy rollouts, and the solution with the highest expected reward is selected for real-world execution. Increasing the noise scale favors robust solutions, while decreasing it permits more precise but higher-risk arrangements.

C. Real-Robot Execution

Once we have a goal pose for each object, and the order in which to place objects, we execute the rearrangement on the real robot. First, we observe the scene from the original camera view, and use Foundation Pose to estimate the latest pose of each object. Then, for each object, we sample grasps using the geometry of the generated mesh. Specifically, we uniformly sample across the triangles of the mesh to determine a contact point [40], and then sample a gripper orientation based on the surface normal at that point. Since we have complete meshes for each object, we can eliminate grasps which are in collision with the environment. We then sample collision-free motion plans for picking and placing the object from its initial pose into its goal pose.

We use the cuRobo library [41] allowing us to perform thousands of collision checks in parallel using CUDA acceleration. For tasks such as stacking, the placement must be precise, and our pose estimation may have error as it is tracking against a generated, imperfect mesh. Therefore, for the final portion of the placement trajectory, we use the force sensor on the end-effector to move the object downward until a sufficient increase in force is detected, indicating that contact has been made with the supporting surface, and the object can be released. This closes the loop on the pre-planned trajectory to avoid dropping the object early.

V. EXPERIMENTS

A. Research Questions

We explore the following research questions: (Q1) How effective is our framework at spatial reasoning, compared with the web-scale prior of vision-language models? (Q2) How important is the 3D prior in real2sim for downstream task success? (Q3) Do solutions from our simulation-based method transfer to the real world? (Q4) How do our design decisions affect performance? (Q5) How accurate and complete are the generated meshes?

B. Experimental Setup

Robot system. To evaluate our framework end-to-end on a real robot system, we use a 7-DoF Franka Panda robot equipped with a compliant gripper and an RGB-D wrist camera (RealSense D435), which captures an image of the scene from a given pose, for input to our real2sim pipeline.
Evaluation tasks. We evaluate our method on a set of challenging spatial reasoning tasks using everyday household objects (Fig. 1). Note that the robot must perform these tasks on these novel objects without human demonstrations or a training dataset of arrangements.
Stacking: the robot must stack the objects (a camera box, a glass bottle, and a roll of tape) into the tallest tower possible. The (x, y) position of the stack is fixed, and the robot must optimize the order in which the objects are placed, as well as their z positions (i.e. height above the table), and their orientations about the x and y axes (where the $+y$ direction faces left from the robot's base frame). This task requires careful long-horizon reasoning about structural stability, spatial creativity in reorienting objects, accurate geometric reconstruction of surfaces, and precise placement during execution. For this task, the reward function is defined to be the height of the tallest point in the scene above the table. The reward is always awarded at the end of the episode, after all objects have been placed and the scene has settled.
Cantilever: there is a large support box on the table, which cannot be moved. The robot can move two objects: a heavy smoothie bottle and a baking paper box. The task is to build the longest structure possible extending out from the support box over the table, without the structure falling and touching the table surface. The x position of the structure is fixed. The robot must optimize the order in which the objects are placed, as well as their z and y positions and orientations about the z axis, to build a structure which maximizes the length of the cantilever while maintaining balance. In order to make this task feasible for a unimanual system, we choose a large support box which is weighed down and has high-friction tape applied to its edge. The reward function is defined as the length of the structure built out from the support box, and if any object touches the table the reward is set to 0.
Dominoes: the robot must knock over the target object (the bottom-heavy, blue tea bottle), using the other objects as "dominoes". Specifically, the robot must choose the x, y, and z positions of a tall green bottle (within the scene bounds of 70 cm $\times$ 10 cm $\times$ 70 cm from the scene origin), and a heavy tin can, as well as their placement order. There is a static pile of objects including a book which cannot be moved by the robot, but can be used as a ramp by placing other objects on top of it. This task requires spatial creativity, and reasoning about dynamic object-object interactions. The reward function is such that the further the target object has been moved from its original pose after the scene has settled, the higher the reward.

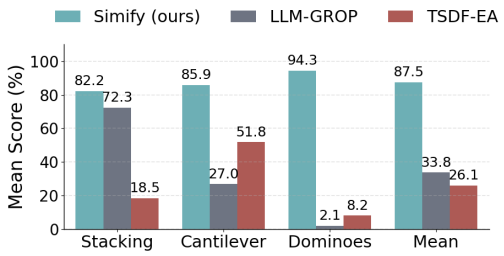

Fig. 3. Normalized task scores for Simify, LLM-GROP, and TSDF-EA. LLM-GROP uses a VLM for goal-pose reasoning, while TSDF-EA uses partial TSDF geometry. Simify consistently finds higher-scoring arrangements.

C. Evaluating Spatial Reasoning in Simulation

In this section we address research questions Q1 and Q2, by comparing our method against the following baselines:
1) LLM-GROP [28]: a recent TAMP framework that uses large language models to infer goal poses for object rearrangement tasks. LLM-GROP is a particularly relevant baseline for our setting because it performs zero-shot goal-pose reasoning without requiring task-specific demonstrations or policy training. It therefore represents a strong test-time planning approach, and directly addresses the question of whether high-level semantic reasoning from an LLM is sufficient for solving our spatial rearrangement tasks. To ensure that the baseline has access to the same input information as our method, the prompt to the language model (GPT-5) includes the reward function, any axis constraints, the image of the scene annotated with the object and scene coordinate frames, and a text scene description including object names, masses, and 3D bounding boxes (obtained using our real2sim pipeline). The LLM is resampled until a parsable output with object order and 3D poses is produced.
2) TSDF-EA [42]: The TSDF-EA baseline (Truncated Signed Distance Field – Evolutionary Algorithm) ablates our use of a 3D prior in the real2sim pipeline: to obtain the mesh for each object, this baseline uses the input RGB-D image to reconstruct the visible geometry via TSDF [42], and then uses the same evolutionary algorithm as our method once in simulation.

We evaluate each method's spatial reasoning solution in simulation. We conduct 10 repeats for each task and for each method, capturing a new image of the initial scene (varying the object poses by a small amount, since they must all fit in the same single view), and then running the full reconstruction and arrangement optimization pipelines, and saving the goal poses output from each method. We then load this solution into an evaluation simulation. Each object

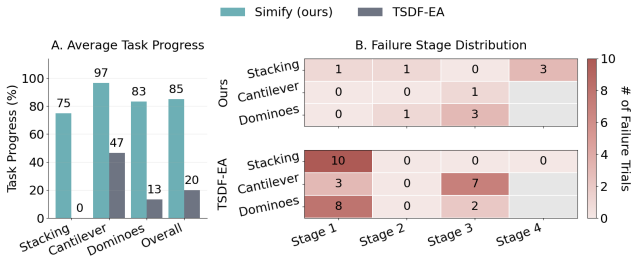

Fig. 4. Real-robot results over 10 trials per task. (A) Mean task progress. (B) Failure-stage distribution, where Stage 1 is planning and later stages are sequential object placements. Simify outperforms TSDF-EA and more often reaches later execution stages.

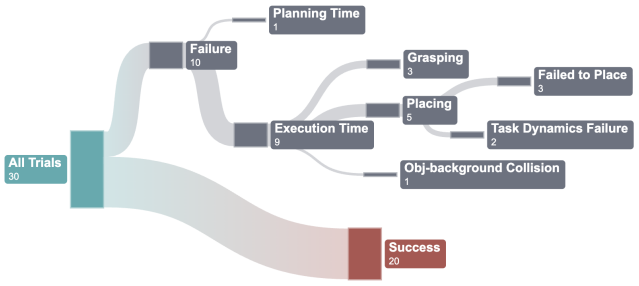

Fig. 5. Failure breakdown for Simify over 30 real-world trials.

is placed into the goal pose determined by the method, with placement noise added, and after the scene settles, we assign a score using the reward function for that task. Note that to evaluate each solution, we use the same mesh in simulation as the mesh used by the method to find the solution during optimization: sim2real transfer is studied in the next section. As we are using simulation, we conduct 2048 evaluation repeats per solution. To allow aggregation across tasks, we rescale the reward into a task score (%) by normalising between a minimum and maximum reward for each task.

The results are shown in Fig. 3. Our method outperforms TSDF-EA on all tasks, showing the importance of using 3D priors in real2sim for generating complete meshes, particularly for tasks where accurately reconstructing unobserved surfaces is important for task success: for example, although the bottom surface of the tin can is in contact with the table and thus cannot be directly observed, our use of image-to-3D models allows this to still be reconstructed accurately, enabling the can to roll down the ramp correctly in simulation. The LLM-GROP baseline performs well on *Stacking*, but is not able to reliably complete the others, whereas our method achieves a higher score on each task. While the VLM understands the high-level semantics of a task such as *Stacking*, it is less capable of determining the continuous pose at which to place each object, which is necessary for tasks requiring precise reasoning about object-object interaction, such as *Cantilever* or *Dominoes*. Our use of a physics simulator as a spatial reasoning engine during deployment enables our method to iteratively refine the poses and complete more precise tasks.

D. Robustness of Real-World Execution

We address Q3 by evaluating Simify end-to-end on real-robot hardware including execution. We compare with TSDF-EA to study the effect of mesh completion on sim2real transfer. For all methods, we terminate a rollout as a planning failure if the selected goal poses achieve a simulation success rate below a fixed threshold, indicating that the arrangement is unstable or unlikely to transfer.

Fig. 4 summarizes real-world task progress and failure stages. Simify achieves substantially higher average task progress than TSDF-EA across all three tasks. The failure-stage distribution further shows that TSDF-EA often fails early, especially in *Stacking*, where all 10 trials fail at Stage 1 (i.e., the planning stage). This indicates that partial geometry from the visible point cloud is often insufficient for discovering stable and executable arrangements. In contrast, the complete meshes generated by Simify lead to fewer planning failures and allow the robot to make progress to later execution stages. In addition, Fig. 5 provides a detailed breakdown of the outcomes for Simify.

TABLE I

TASK SCORE (%) RESULTS FOR AN ABLATION STUDY, EXPLORING THE DESIGN CHOICES IN OUR EVOLUTIONARY ALGORITHM.

Method	*Stacking*	*Cantilever*	*Dominoes*	Mean
MeshGen-Rand	29.8	0.0	1.1	10.3
Simify-NoVal	62.6	45.6	83.8	64.0
Simify (ours)	**82.2**	**85.9**	**94.3**	**87.5**

E. Ablating Design Choices for Evolutionary Optimization

To answer Q4, we ablate the main design choices in our evolutionary optimization algorithm. Table I compares Simify against two variants. MESHGEN-RAND uses the same generated meshes from our real2sim pipeline, but replaces our evolutionary optimization and validation algorithm with randomly sampled object poses and placement orders. Its low performance shows that complete geometry alone is not sufficient: the tasks require explicit spatial reasoning over the rearrangement sequence and continuous object poses. SIMIFY-NOVAL removes the final validation stage, where candidate solutions are evaluated across multiple noisy rollouts to estimate expected reward under placement error. This variant performs better than random search, but substantially worse than Simify, showing that selecting solutions based only on their best optimization score can produce arrangements that are brittle to execution noise (see Fig. 6 for an illustrative example). In contrast, our validation stage favors robust arrangements that remain successful under perturbations.

We further study how performance scales with the number of parallel simulation environments. Fig. 7 shows that increasing the number of environments improves task score across all three tasks. This trend is especially clear for *Cantilever* and *Dominoes*, where larger populations help the optimizer discover more precise and physically robust arrangements. These results indicate that Simify can effectively use parallel compute during test-time reasoning: more simulation environments provide broader exploration of the mixed

Fig. 6. *Dominoes* solutions. **Left:** A high-risk strategy that places the green bottle on the can to tip it toward the target; it occasionally succeeds but is sensitive to placement noise. **Right:** The selected robust strategy, which places the bottle between the objects and rolls the can down the ramp. Validation favors the solution with higher expected reward.

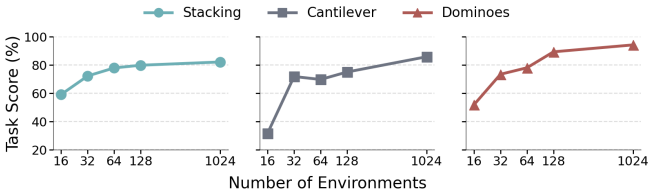

Fig. 7. Effect of simulation parallelism on task score. More environments improve performance on all tasks, showing that Simify benefits from additional test-time compute.

discrete-continuous solution space and lead to higher-quality solutions. With our default setting of 1024 environments, optimization takes only 73 seconds on average, making the approach practical for real-robot deployment.

F. The Effect of Mesh Quality on Task Performance

TABLE II

SURFACE RECONSTRUCTION QUALITY OF GENERATED AND PARTIAL TSDF MESHES, MEASURED AGAINST GROUND-TRUTH YCB MESHES.

			Chamfer Completeness ↓	Chamfer Accuracy ↓	F1-Score ↑	Normal Consistency ↑
Low Occ.	TSDF [42]	Bottle	0.015	0.003	0.687	0.752
		Bowl	0.009	0.003	0.801	0.787
		Box	0.026	0.002	0.596	0.791
		Pitcher	0.022	0.002	0.642	0.728
		Mean	0.018	**0.002**	0.682	0.765
	Mesh Gen. (ours)	Bottle	0.004	0.004	0.939	0.917
		Bowl	0.002	0.003	0.996	0.904
		Box	0.003	0.003	0.975	0.945
		Pitcher	0.005	0.007	0.797	0.840
		Mean	**0.004**	0.004	**0.927**	**0.902**
High Occ.	TSDF [42]	Bottle	0.016	0.002	0.656	0.756
		Bowl	0.018	0.003	0.602	0.723
		Box	0.036	0.003	0.457	0.749
		Pitcher	0.024	0.002	0.600	0.722
		Mean	0.024	**0.002**	0.579	0.738
	Mesh Gen. (ours)	Bottle	0.004	0.004	0.986	0.907
		Bowl	0.002	0.004	0.967	0.872
		Box	0.009	0.010	0.556	0.778
		Pitcher	0.005	0.008	0.790	0.838
		Mean	**0.005**	0.007	**0.825**	**0.849**

To answer Q5, we compare our mesh generation pipeline against the TSDF partial meshes using reconstruction metrics. We run our real2sim pipeline on scenes composed of YCB objects [43], for which ground-truth meshes are available. We align the estimated mesh against the ground-truth mesh using ICP. Chamfer Completeness measures the Euclidean distance from each sampled point on the ground-truth mesh to the nearest point on the estimated mesh, while Chamfer Accuracy measures the distance from each point on the estimated mesh to the nearest on the ground-truth mesh. The F1-Score is the harmonic mean of precision and recall,

Fig. 8. Qualitative comparison of partial TSDF, SAM3D, and Hunyuan3D-2 meshes at different octree resolutions. We use Hunyuan3D-2 at resolution 64 to balance reconstruction speed and accuracy; the modular pipeline also supports alternative image-to-3D models.

calculated as the proportion of points satisfying a 0.01 m distance threshold. Normal Consistency uses the dot-product between the mesh surface normals.

Results in TABLE II show that while the depth-based TSDF meshes are marginally more accurate on visible surfaces, the generated meshes are significantly more complete due to the powerful 3D prior used, and so have a lower Chamfer Completeness distance, as well as a higher overall F1-Score and Normal Consistency. Our mesh generation pipeline is also more robust to occlusions than the baseline which does not use visual priors. While we leave an in-depth comparison of image-to-3D models on standard benchmarks to prior work [44], this shows that for our spatial reasoning problem setting, geometric completeness is important for downstream task performance and sim2real transfer.

VI. CONCLUSIONS, LIMITATIONS, AND FUTURE WORK

We present Simify, a framework for solving spatial reasoning tasks involving multi-object interaction, by first building a simulation of the scene using the powerful visual prior of image-to-3D models, and then discovering object arrangements which solve the task through evolutionary search using massively parallel physics simulation at inference time. Results show the importance of complete and accurate geometry for downstream task success, and demonstrate how our framework can perform multi-stage spatial reasoning tasks with novel objects and no training arrangements required.

In this work, we focus on the role of geometry in real2sim for spatial reasoning, and assume that physics parameters such as mass and friction are given. We assume uniform density when the center of mass is set, and find that our method is sensitive to these parameters (in particular, in *Cantilever*). Automatically inferring these is an active area of research [11], [13]. Future work can incorporate these techniques, and could also extend our framework beyond rigid objects. Another limitation of our method is that in order to simplify and accelerate simulation optimization, we place objects directly into their placement poses without a simulated robot. While we do use heuristics to filter out kinematically infeasible solutions (e.g. not sampling rotations directly away from the robot), the rearrangement may not always be feasible to execute directly. While most recent image-to-3D models use single-view input (Fig. 8), and our framework inherits their limitations including precision [44], future image-to-3D models which take advantage of multi-view input can be swapped into our framework. Finally, combining our test-time simulation-based method with a VLM for high-level reasoning [45] is a promising future work direction: the VLM could consider object semantics

(our simulation only considers geometry), set the constraints for the optimization, and propose initial guesses to accelerate planning, particularly for larger scenes.

REFERENCES

[1] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, "Lerf: Language embedded radiance fields," in *International Conference on Computer Vision (ICCV)*, 2023.

[2] A. Majumdar, A. Ajay, and X. e. a. Zhang, "Openeqa: Embodied question answering in the era of foundation models," in *CVPR*, 2024.

[3] Y. Hong, H. Zhen, P. Chen, S. Zheng, Y. Du, Z. Chen, and C. Gan, "3d-llm: Injecting the 3d world into large language models," *NeurIPS*, 2023.

[4] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," in *CoRL*, 2023.

[5] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa *et al.*, "Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning," in *ICRA*, 2024.

[6] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai *et al.*, "$\pi_{0.5}$: a vision-language-action model with open-world generalization," *CoRL*, 2025.

[7] NVIDIA, "GR00T N1: An open foundation model for generalist humanoid robots," in *ArXiv Preprint*, March 2025.

[8] J. Lee, J. Duan, H. Fang, Y. Deng, S. Liu, B. Li, B. Fang, J. Zhang, Y. R. Wang, S. Lee, W. Han, W. Pumacay, A. Wu, R. Hendrix, K. Farley, E. VanderBilt, A. Farhadi, D. Fox, and R. Krishna, "Molmoact: Action reasoning models that can reason in space," 2025.

[9] C. He, X. Liu, G. S. Camps, J. Bruno, G. Sartoretti, and M. Schwager, "Demystifying robot diffusion policies: Action memorization and a simple lookup table alternative," in *ICLR*, 2026.

[10] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. D. Ratliff, and D. Fox, "Closing the sim-to-real loop: Adapting simulation randomization with real world experience," *ICRA*, 2019.

[11] M. Memmel, A. Wagenmaker, C. Zhu, P. Yin, D. Fox, and A. Gupta, "ASID: Active exploration for system identification in robotic manipulation," *ICLR*, 2024.

[12] B. Bianchini, M. Halm, and M. Posa, "Simultaneous learning of contact and continuous dynamics," in *CoRL*, 2023.

[13] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg, "Real2sim2real: Self-supervised learning of physical single-step dynamic actions for planar robot casting," in *ICRA*, 2022.

[14] Z. Chen, A. Walsman, M. Memmel, K. Mo, A. Fang, K. Vemuri, A. Wu, D. Fox, and A. Gupta, "URDFormer: A pipeline for constructing articulated simulation environments from real-world images," *RSS*, 2024.

[15] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, "Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation," *RSS*, 2024.

[16] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Suenderhauf, "Physically embodied gaussian splatting: A realtime correctable world model for robotics," in *CoRL*, 2024.

[17] S. Patel, X. Yin, W. Huang, S. Garg, H. Nayyeri, L. Fei-Fei, S. Lazebnik, and Y. Li, "A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards," in *ICRA*, 2025.

[18] A. Agarwal, G. Singh, B. Sen, T. Lozano-Pérez, and L. P. Kaelbling, "SceneComplete: Open-world 3D scene completion in complex real world environments for robot manipulation," *ArXiv*, 2024.

[19] W. Ye, F. Liu, Z. Ding, Y. Gao, O. Rybkin, and P. Abbeel, "Video2Policy: Scaling up manipulation tasks in simulation through internet videos," *ArXiv*, 2025.

[20] H. Jiang, H.-Y. Hsu, K. Zhang, H.-N. Yu, S. Wang, and Y. Li, "PhysTwin: Physics-informed reconstruction and simulation of deformable objects from videos," *ICCV*, 2025.

[21] Y. Wang, Z. Xian, F. Chen, T.-H. Wang, Y. Wang, K. Fragkiadaki, Z. Erickson, D. Held, and C. Gan, "Robogen: Towards unleashing infinite data for automated robot learning via generative simulation," *ICML*, 2024.

[22] P. Katara, Z. Xian, and K. Fragkiadaki, "Gen2Sim: Scaling up robot learning in simulation with generative models," *ICRA*, 2024.

[23] Y. Mu, T. Chen, Z. Chen, S. Peng, Z. Lan, Z. Gao, Z. Liang, Q. Yu, Y. Zou, M. Xu, L. Lin, Z. Xie, M. Ding, and P. Luo, "RoboTwin: Dual-arm robot benchmark with generative digital twins," in *CVPR*, 2025.

[24] I. Kapelyukh, X. Zhang, S. James, L. Herlant, and E. Johns, "ZeroBot: Learning from scratch in minutes with generative real2sim," *IEEE RA-L*, 2026.

[25] D. Batra, A. X. Chang, S. Chernova, A. J. Davison, J. Deng, V. Koltun, S. Levine, J. Malik, I. Mordatch, R. Mottaghi, M. Savva, and H. Su, "Rearrangement: A challenge for embodied ai," *ArXiv*, 2020.

[26] A. Curtis, X. Fang, L. P. Kaelbling, T. Lozano-Pérez, and C. R. Garrett, "Long-horizon manipulation of unknown objects via task and motion planning with estimated affordances," in *ICRA*, 2022.

[27] Z. Yang, J. Mao, Y. Du, J. Wu, J. B. Tenenbaum, T. Lozano-Pérez, and L. P. Kaelbling, "Compositional Diffusion-Based Continuous Constraint Solvers," in *Conference on Robot Learning*, 2023.

[28] X. Zhang, Y. Ding, Y. Hayamizu, Z. Altaweel, Y. Zhu, Y. Zhu, P. Stone, C. Paxton, and S. Zhang, "Llm-grop: Visually grounded robot task and motion planning with large language models," *The International Journal of Robotics Research*, p. 02783649251378196, 2025.

[29] X. Zhang, Z. Altaweel, Y. Hayamizu, Y. Ding, S. Amiri, H. Yang, A. Kaminski, C. Esselink, and S. Zhang, "Dkprompt: Domain knowledge prompting vision-language models for open-world planning," *arXiv preprint arXiv:2406.17659*, 2024.

[30] W. Liu, C. Paxton, T. Hermans, and D. Fox, "Structformer: Learning spatial structure for language-guided semantic rearrangement of novel objects," in *ICRA*, 2022.

[31] W. Liu, Y. Du, T. Hermans, S. Chernova, and C. Paxton, "StructDiffusion: Language-guided creation of physically-valid structures using unseen objects," in *RSS*, 2023.

[32] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar, "Eureka: Human-level reward design via coding large language models," in *ICLR*, 2024.

[33] R. Gong*, X. Zhang*, J. Shang*, M. V. Minniti*, J. Patel, V. Pepe, R. Yan, A. Gundogdu, I. Kapelyukh, A. Abbas, X. Yan, H. Patel, L. Herlant, and K. Schmeckpeper, "AnyTask: an automated task and data generation framework for advancing sim-to-real policy learning," *arXiv preprint arXiv:2512.17853*, 2025.

[34] S. Pan, L. Jin, X. Huang, C. Stachniss, M. Popović, and M. Bennewitz, "DM-OSVP++: One-shot view planning using 3d diffusion models for active rgb-based object reconstruction," 2025.

[35] N. Ravi, V. Gabeur, Y.-T. Hu, R. Hu, C. Ryali, T. Ma, H. Khedr, R. Rädle, C. Rolland, L. Gustafson, E. Mintun, J. Pan, K. V. Alwala, N. Carion, C.-Y. Wu, R. Girshick, P. Dollár, and C. Feichtenhofer, "SAM 2: Segment anything in images and videos," *ArXiv*, 2024.

[36] Tencent Hunyuan3D Team, "Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3D assets generation," 2025.

[37] A. Ardelean, M. Özer, and B. Egger, "Generalizable 3D scene reconstruction via divide and conquer from a single view," in *International Conference on 3D Vision (3DV)*, 2025.

[38] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[39] Nvidia, "Isaac Lab: A GPU-Accelerated Simulation Framework for Multi-Modal Robot Learning," NVIDIA Technical Report, 2025.

[40] R. Osada, T. Funkhouser, B. Chazelle, and D. Dobkin, "Shape distributions," *ACM Transactions on Graphics*, 2002.

[41] B. Sundaralingam, S. K. S. Hari, A. Fishman, C. Garrett, K. V. Wyk, V. Blukis, A. Millane, H. Oleynikova, A. Handa, F. Ramos, N. Ratliff, and D. Fox, "cuRobo: Parallelized collision-free minimum-jerk robot motion generation," *ArXiv*, 2023.

[42] W. Dong, Y. Lao, M. Kaess, and V. Koltun, "Ash: A modern framework for parallel spatial hashing in 3d perception," *IEEE TPAMI*, 2021.

[43] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols," *IEEE RAM*, 2015.

[44] F. Nolte, A. Geiger, B. Schölkopf, and I. Posner, "Is single-view mesh reconstruction ready for robotics?" *ArXiv*, 2025.

[45] H. Liu, S. Yao, H. Chen, J. Gao, J. Mao, J.-B. Huang, and Y. Du, "SIMPACT: Simulation-enabled action planning using vision-language models," in *CVPR*, 2026.