Building Explicit World Model for Open-world Object Manipulation

Xiaotong Li

Delft University of Technology

TU Delft

Building Explicit World Model for Open-world Object Manipulation

by

Xiaotong Li

5965373

M.Sc. Robotics

Main supervisor:	Dr. J. (Javier) Alonso Mora
Daily supervisor:	Dr. Clarence Chen
Duration:	January, 2025 - October, 2025
Faculty:	Cognitive Robotics (CoR), Mechanical Engineering, Delft

Cover: Generated by GPT-5
Style: TU Delft Report Style, with modifications by Xiaotong Li

TU Delft

Abstract

Open-world object manipulation has emerged as a popular research frontier in robotics. While recent advances in vision-language-action (VLA) models have achieved impressive results, they typically rely on large amounts of task-specific action data for training. This thesis aims to enable a manipulator to perform open-world object manipulation tasks without any action demonstrations. Instead of learning direct action mappings, we focus on understanding object dynamics. To this end, we propose a novel framework that builds an explicit world model for open-world object manipulation. The framework integrates open-set segmentation and grasping, 3D digital twin reconstruction, and simulation-based strategy sampling within a unified framework. At the core of our approach lies the construction of a physically grounded digital twin of the environment, which enables the framework to simulate and evaluate diverse interaction strategies before real-world execution. Experimentally, the proposed framework is able to perform multiple open-set manipulation tasks, such as “put the banana into the basket”, “stack the green cube onto the yellow cube”, and “place the blue cup upside on the wooden box”. These results are obtained without any task-specific action demonstrations, demonstrating strong generalization and autonomy compared to existing closed-set or imitation-based systems.

Contents

Abstract i

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Problem Statement . . . 2
- 1.3 Contribution . . . 2
- 1.4 Document Structure . . . 3

2 Related Works 4
- 2.1 Grasping Pose Prediction . . . 4
- 2.2 Open-set Detection and Segmentation . . . 5
- 2.3 Vision-Language-Action Models For Robotics . . . 6
- 2.4 World Model . . . 7
- 2.5 Motion Planning for Manipulation . . . 8

3 Methods 10
- 3.1 System Overview . . . 10
- 3.2 Open-set Segmentation . . . 10
- 3.3 Open-set Object Grasping . . . 12
  - 3.3.1 Grasp pose predictor . . . 12
  - 3.3.2 Grasp pose selector . . . 12
  - 3.3.3 Grasp result checker . . . 12
- 3.4 Digital twin Construction . . . 13
  - 3.4.1 Object mesh generation . . . 13
  - 3.4.2 Object mesh alignment . . . 13
  - 3.4.3 Material property prediction . . . 15
- 3.5 Manipulation Strategy Sampling in Simulation . . . 15
  - 3.5.1 Interaction area segmentation . . . 16
  - 3.5.2 Interaction strategy sampling . . . 16
  - 3.5.3 Result checking and action selection . . . 17

4 Experiments 20
- 4.1 Hardware Setup . . . 20
- 4.2 Software Setup . . . 21
- 4.3 Digital Twin Construction Accuracy . . . 21
- 4.4 Real Robot Task Performance . . . 22
  - 4.4.1 Experiment setup . . . 22
  - 4.4.2 Quantitative results . . . 24

5 Conclusion and Future Works 28
- 5.1 Conclusion . . . 28
- 5.2 Limitation and Future Works . . . 28

References 30

1 Introduction

1.1. Background

Open-world object manipulation has recently become a popular research frontier in robotics, aiming to enable robots to perform diverse tasks commanded by humans in unstructured environments. In such settings, robots must have the ability to perceive and reason about previously unseen objects, infer task goals from natural language, and interact physically with the world, without relying on pre-defined categories or manually curated policies. This open-ended nature introduces fundamental challenges in semantic and physic understanding.

Recent advances in large language models (LLMs) and vision-language models (VLMs) have demonstrated remarkable capabilities in factual reasoning, conceptual understanding, and commonsense inference [1, 2, 3, 4], enabled by large-scale pretraining on Internet-scale multimodal data. Despite their impressive generalization across perception and language tasks, there remains a substantial gap between these foundation models and the requirements of embodied robotic systems. Specifically, LLMs and VLMs still lack a grounded understanding of physical dynamics and low-level robot control, which are essential for translating abstract task instructions into executable manipulation behaviors.

To bridge this gap, recent studies have explored Vision-Language-Action (VLA) models [5, 6, 7, 8, 9], which leverage large-scale vision-language models to infer the robotic actions for different tasks. VLA-based systems have demonstrated impressive performance across a variety of manipulation tasks, showing the potential of integrating multimodal reasoning with embodied control. However, training such models remains non-trivial, as they require vast amounts of task-specific action data, which are costly to collect. Furthermore, the learned policies often need fine-tuning when transferred to robots with new embodiments or sensing configurations, limiting their generalization to truly open-world settings.

To overcome these limitations, a growing body of research has turned to the concept of a world model—a representation that allows reasoning about how the environment reacts and evolves under different agent interactions. Instead of directly mapping sensory inputs to motor commands, world-model-based approaches aim to learn or construct a model of the world's dynamics, enabling prediction, simulation, and planning in a physically grounded manner. This paradigm shifts the focus from learning to act to learning how the world works, which offers stronger generalization to unseen objects, tasks, and environments.

With the growing computational power thanks to the hardware development, we see recent image-based and video-based world models [10, 11, 12, 13, 14] leverage the inherent world knowledge of video diffusion models [15, 16] to understand the temporal and spatial relationships of the generated world. However, while these models can generate visual compelling results, they inherently lack 3D consistency, due to the 2D training data and underlying 2D representation.

In contrast, explicit world models represent the 3D environment through geometrically and physically meaningful components, such as 3D object meshes. By leveraging physically grounded simulators,

which inherently encode real-world dynamics such as gravity, friction, and collision response, these explicit representations enable dynamic reasoning and interaction simulation, making them particularly suitable for real-world manipulation and task planning. Given that some existing simulators can already provides reliable physical properties [17], the remaining challenge lies in constructing an accurate digital twin of the real-world scene — that is, building a digital twin that faithfully captures the geometry, pose, and other physical information (e.g. material) of the observed objects. Once such a digital twin is established, the robot can use it to simulate, evaluate, and select feasible interaction strategies before execution in the physical world.

1.2. Problem Statement

In an open-world object manipulation task, the set of objects and tasks is not fixed or predefined. Unlike structured industrial environments—where the robot interacts with known objects under well-calibrated conditions—an open-world setting involves previously unseen objects, diverse spatial configurations, and natural language task instructions that can vary in form and complexity. In such a setting, a robot must not only detect and localize arbitrary objects, but also interpret task intent and reason about the physical consequences of potential actions.

Formally, given the visual observations I of the scene (e.g., RGB-D images), and a natural language task instruction C, and a world model W, the objective is to sample a set of executable manipulation actions a, evaluate them and find an a^* that successfully achieves the goal described by C in the real world. That is,

$$W : p_W(\tau \mid I, a), \tag{1.1}$$

where $\tau = (s, o)$ is the future states and observations generated by world model W, given the current observation I and sampled action a. Then,

$$a^* = \arg\max_{a \in \mathcal{A}(\mathcal{I})} E_{\tau \sim p_W(\cdot|I,a)}[R(C, \tau)], \quad \mathcal{A} = \{a_i\} \sim \pi(\cdot \mid I, C), i = 1, ..., N, \tag{1.2}$$

where $\mathcal{A}$ denotes the space of sampled candidate actions from a sample policy $\pi(\cdot \mid I, C)$, parameterized by end-effector poses, and $R(C, \tau)$ is a score function that measures the score of τ given the natural language task instruction C.

Thus, in this thesis, we focus on how to build an explicit world model that can generate plausible future states and observations τ. We also designed a prior-based sample policy π and a VLM-based result checker to measure the success score of τ.

Besides, the scope of this work is limited to manipulation tasks involving rigid objects, as deformable and articulated objects exhibit significantly more complex dynamics and require additional modeling of elasticity, joint constraints, and contact deformation. These aspects are beyond the scope of this thesis and are left for future research.

1.3. Contribution

This thesis addresses the above challenge by constructing an explicit, physically grounded world model from visual observations, enabling the robot to reason about object dynamics and to predict and evaluate the success probability of candidate actions. We present a novel explicit-world-model-based framework for performing manipulation tasks in open-set environments that involve previously unseen objects and tasks, without requiring any task demonstrations or tele-operated action data.

While prior work such as [18] explores a related concept of constructing an explicit world model within a high-fidelity simulator and evaluating action candidates for real robots, their approach is restricted to single articulated-object manipulation and the task goal focuses solely on achieving target articulation angles. In contrast, our work tackles the more general problem of open-set manipulation task involving multiple objects and diverse interaction goals.

The contribution of this thesis are listed as follows:

- An explicit-world-model-based manipulation framework that integrates open-set object segmentation and grasping, 3D digital twin reconstruction, and simulation-based strategy sampling for open-world object manipulation.
- A dynamic digital twin construction pipeline that generates object meshes from RGB images and aligns their poses to real-world observations to enable real-world-consistent simulation.
- A simulation-guided action evaluation method that estimates the success probability of candidate strategies by reasoning over physically simulated outcomes.
- We demonstrate the system's effectiveness across diverse open-set manipulation tasks involving multiple unseen objects on a real robot.

1.4. Document Structure

The remainder of this thesis is organized as follows: Chapter 2 reviews literature related to four key research areas: grasping pose prediction, open-set detection and segmentation, vision-language-action models, world models and motion planning for manipulators. Chapter 3 introduces the overall framework proposed in this thesis and provides a detailed description of the methodology we use. Chapter 4 presents the experiment results, where we examine the digital twin construction accuracy and real-world tasks performance, and present our analysis. Finally, Chapter 5 concludes the thesis by discussing the findings, identifying limitations, and outlining potential directions for future research.

2

Related Works

This chapter reviews existing research related to the proposed framework. We begin with grasp pose prediction, which directly corresponds to the grasping module adopted in this work. Next, we discuss open-set object detection and segmentation, which enables the perception of arbitrary objects beyond predefined categories. We then review recent progress in Vision-Language-Action (VLA) models, highlighting their strengths and limitations in robotic tasks. Furthermore, we examine advances in world model construction, including both image/video-based and explicit world models. Finally, we discuss existing methods for manipulation planning.

2.1. Grasping Pose Prediction

Analytic grasp planning methods (e.g., [19]) typically adopt a multi-stage pipeline, where an input RGB-D image or point cloud is sequentially processed through several stages, such as segmentation, object classification, and pose estimation. However, errors introduced at any of these intermediate stages can accumulate and significantly degrade the overall grasp success rate.

An alternate approach is to use deep learning to learn grasp representations. Some early work use the form of oriented 2D grasp rectangles [20, 21, 22, 23, 24] based on RGB-D image input. Lenz et al.[20] proposed a two-stage method that a small network is first used to search all potential rectangles and a larger network is then used to find the top-ranked rectangle from these candidates. Fig 2.1 and Fig 2.2 illustrate their pipeline and experiment results.

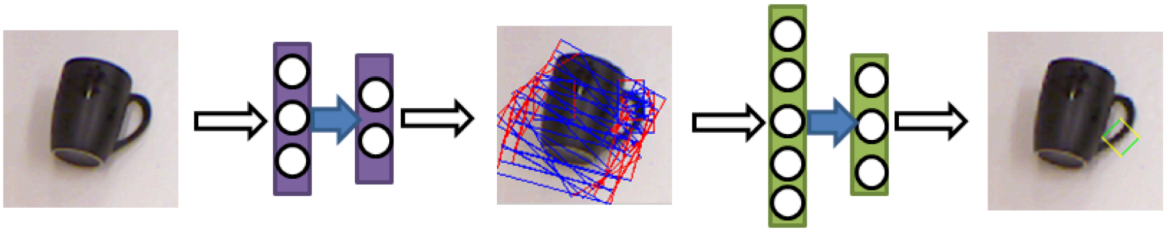

Figure 2.1: The two-stage pipeline from [20].

However, grasp rectangle representations are constrained in the degrees of freedom of the resulting grasp poses, thereby limiting their applicability in more general settings. To address this, subsequent research has focused on predicting full 6-DOF grasp poses. Some studies approach this by evaluating a set of candidate grasps, framing the problem as a sample-and-evaluate pipeline [25, 26, 27, 28], in which grasp candidates are densely and uniformly sampled throughout the scene and then evaluated using a learned grasp quality network. Meanwhile, other researchers propose end-to-end methods [29, 30, 31], where networks are trained to directly process the observed point cloud and output the most feasible 6-DOF grasp poses without explicit candidate sampling. Fig 2.3 illustrates the end-to-end architecture of GSNet from [32].

Figure 2.2: Left: Experiment results with rectangles corresponding to robotic grasps. Green lines correspond to robotic gripper plates. Center: A Baxter robot "Yogi" successfully executing a grasp detected by algorithm in [20]. Right: The grasp detected in the RGB (top) and depth (bottom) images obtained from Kinect.[20]

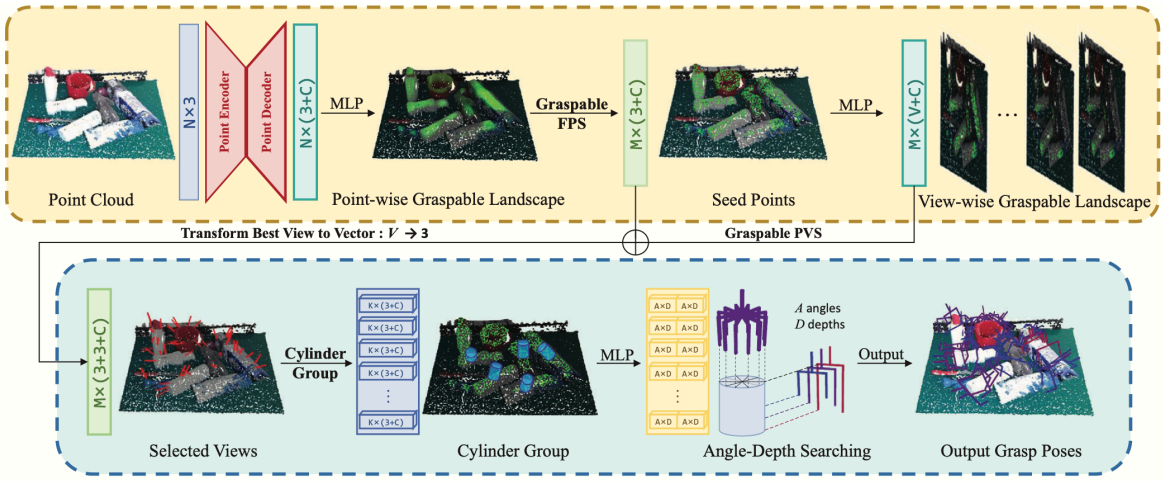

Figure 2.3: GSNet architecture. The upper row show the process of cascaded graspness model and the lower row shows the grasp operation model. In cascaded graspness model, point encoder-decoder outputs C-dim feature vectors for the input N points. A point-wise graspable landscape is generated and M seed points are sampled from it. The seeds are then used to generate view-wise graspable landscapes, and select the grasp view. In grasp operation model, the seeds are grouped in cylinder regions. The grasp scores and gripper widths are predicted for each group and used to output M grasp poses.[32]

2.2. Open-set Detection and Segmentation

Traditional vision-based detection and segmentation tasks are usually restricted to a fixed set of predefined object classes, represented as integer labels. To enable open-set detection, a model must be capable of processing both visual and textual information. Recent advances in vision-language modeling provide promising directions. CLIP [2] was one of the pioneering works in this area. By jointly training an image encoder and a text encoder via contrastive learning, CLIP successfully aligned image features with text features, as shown in Fig 2.4.

Despite its strong visual-textual alignment and generalization ability, CLIP operates only at the whole image level and lacks the capacity to localize or differentiate multiple elements within an image. To address this limitation, Li et al. [33] introduced GLIP, which integrates the ideas of CLIP with traditional object detectors. GLIP first generates region proposals from global image features and then extracts features for each proposal, aligning them with the corresponding text descriptions. As the authors describe, this approach "unifies detection and grounding by reformulating object detection as phrase grounding." Consequently, the detection results are no longer constrained by a fixed label set but can extend to arbitrary textual descriptions, including abstract concepts, thus realizing open-set detection.

Building on this foundation, Grounding-DINO [34] further improves performance by replacing the conventional two-stage detector with the transformer-based detector DINO (DETR with Improved deNoising anchOr boxes) [35]. To advance this line of research, the authors proposed Grounded-SAM [36], a

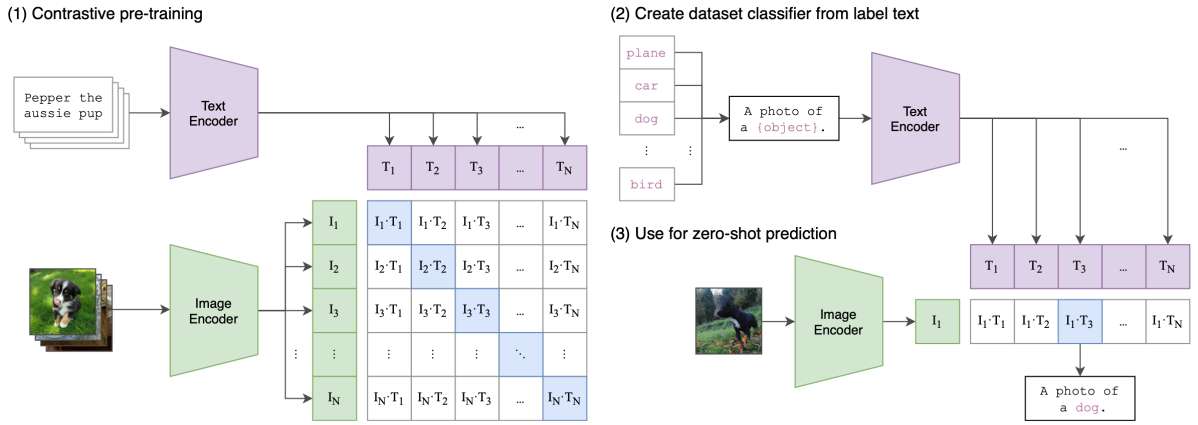

Figure 2.4: The contrastive pre-training approach adopted by CLIP [2]. The authors jointly trained an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples.

compositional framework that combines Grounding-DINO with Segment Anything (SAM) [37], enabling accurate segmentation of target objects in an open-world setting.

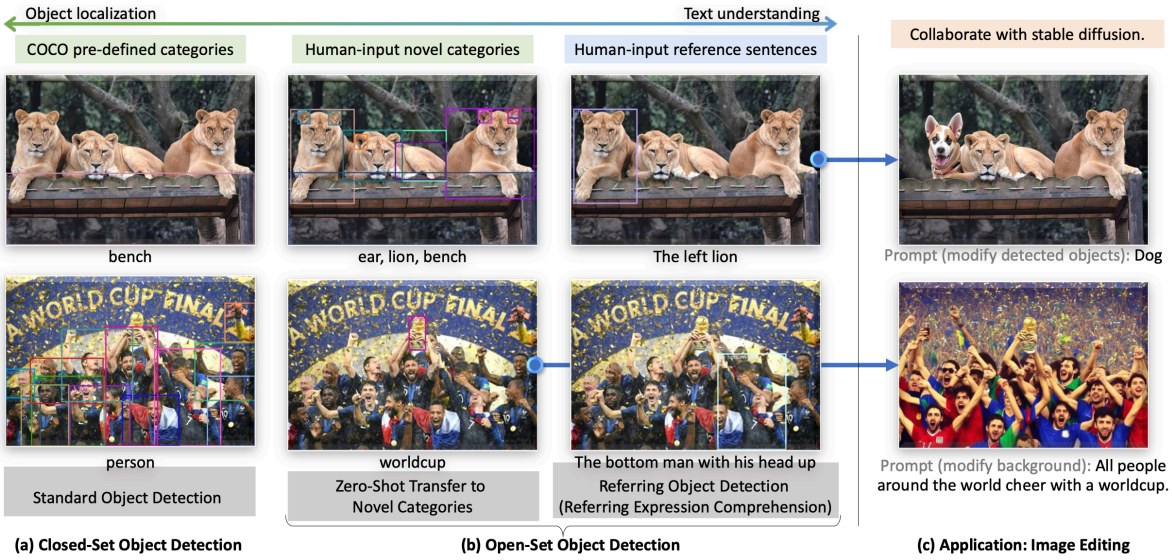

Figure 2.5: Caption

2.3. Vision-Language-Action Models For Robotics

Recent progress in large foundation models—particularly large language models (LLMs) [1, 3, 38, 39, 40]—has led to remarkable advances in natural language understanding, reasoning, and generation. Meanwhile, vision-language models (VLMs) [2, 4, 41] extend these capabilities by jointly modeling visual and textual modalities, enabling multimodal perception, understanding, and generation. Together, these foundation models encode broad world knowledge, demonstrate strong reasoning capabilities, and generalize effectively to novel tasks and scenarios, making them highly adaptable across a wide range of domains.

Despite these achievements, the abilities of LLMs and VLMs remain largely constrained to the digital domain, limiting their direct impact on embodied, real-world tasks. To bridge this gap, recent research has explored leveraging the perceptual and reasoning capabilities of foundation models to guide robotic task execution, effectively extending their intelligence into the physical world. This emerging research direction has given rise to Vision-Language-Action (VLA) models [5, 6, 7, 8, 9], which can be broadly de-

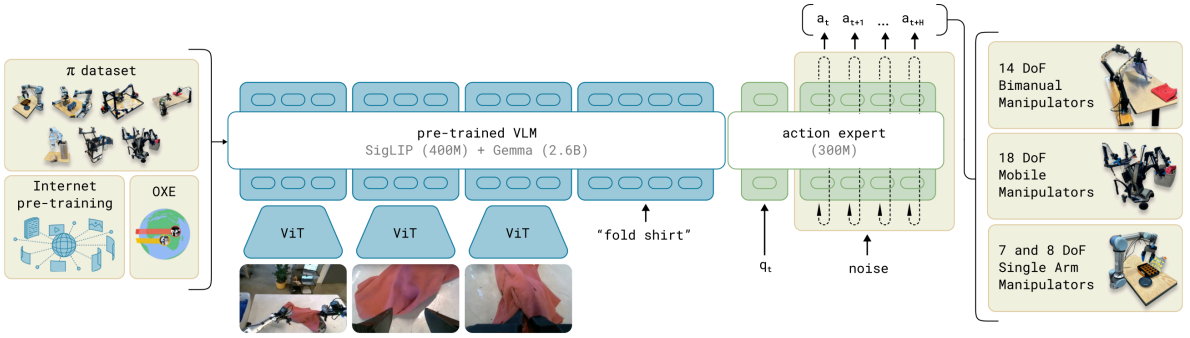

Figure 2.6: Overview of the π_0 framework [7]. The model is trained via a flow-matching VLA architecture that combines a large vision-language backbone with a lightweight action expert for robot state and action generation. Pre-trained on both self-recorded and open-source manipulation datasets, π_0 generalizes across multiple robot embodiments and task types.

fined as systems that generate executable actions conditioned on visual and linguistic inputs, grounded by at least one large-scale vision or language foundation model.

However, training a VLA model is non-trivial. For example, in [7], the authors collected 10,000 hours of dexterous manipulation data from 7 different robot configurations and 68 tasks, in addition to large amounts of previously collected robot manipulation data from OXE [42], DROID [43], and Bridge [44]. This shows that the performance of VLA models depends highly on high-quality task-related demonstrations.

2.4. World Model

Following NVIDIA's definition, world models are "generative AI models that understand the dynamics of the real world, including physics and spatial properties" [45]. In essence, world models summarize an agent's past experiences into a compact predictive model[10]. By encapsulating the environment's dynamics, they allow agents to simulate interactions, plan future actions, and make informed decisions without requiring direct execution in the real world.

Recent advances in world model have focused primarily on image/video-based world models [10, 11, 12, 13, 14], which leverage the inherent world knowledge of video diffusion models [15, 16] to understand the temporal and spatial relationships of the generated world. However, while these models can generate visual compelling results, they inherently lack 3D consistency, due to the 2D training data and underlying 2D representation.

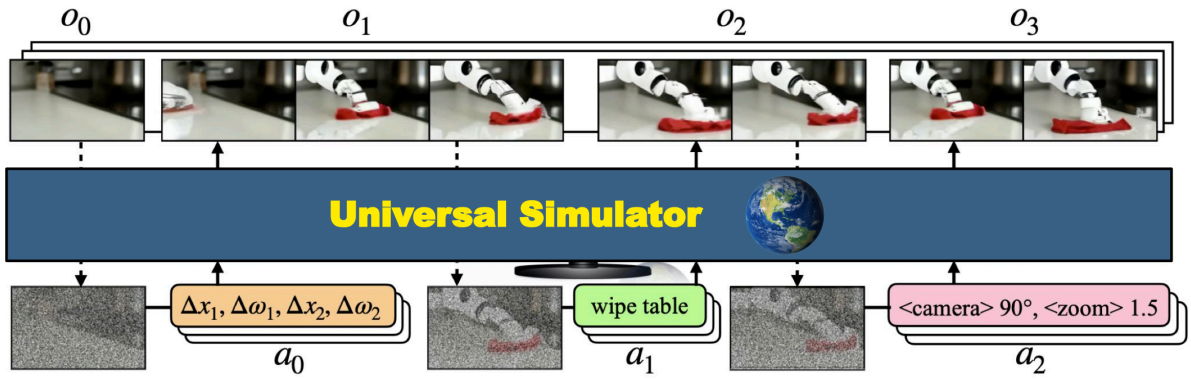

Figure 2.7: Training and inference of a video-based world model [13]. It is a video diffusion model trained to predict the next (variable length) set of observation frames (o_t) given observations from the past (e.g., o_{t1}) and action input a_{t1}.

However, these world models typically require large-scale data and significant computational resources for training . Moreover, their inference speed is often insufficient for real-time applications, thus not well suited for evaluating different action candidates. Currently, they are mostly integrated into end-to-end

learning frameworks to guide action generation, such as Wu et al. [12] employed the Dreamer world model [11] for online learning in four real-world robotic tasks—without relying on any external simulators.

Another line of research focuses on using point cloud completion and 3D AIGC (AI-generated content) techniques to construct explicit world models for the object of interest only. This strategy not only enables interactive planning and long-horizon decision-making but also significantly reduces the computational overhead involved in model construction (only multi-view images instead of videos).

For example, Mohammadi et al.[46] build object-centric explicit world models from point clouds to enable interactive grasp prediction. Similarly, Hidalgo-Carvajal et al.[47] adopt a comparable approach using signed distance fields (SDFs) for dexterous manipulation. Magistri et al.[48] employ a DeepSDF-based framework[49] to reconstruct complete 3D shapes of fruits from partial observations, which are then combined with pose estimation to determine suitable grasp or harvesting poses.

More recently, Jiang et al. [18] construct explicit world models for articulated objects and apply sampling-based model predictive control within a physics simulator to plan trajectories for various tasks—without requiring demonstrations or reinforcement learning.

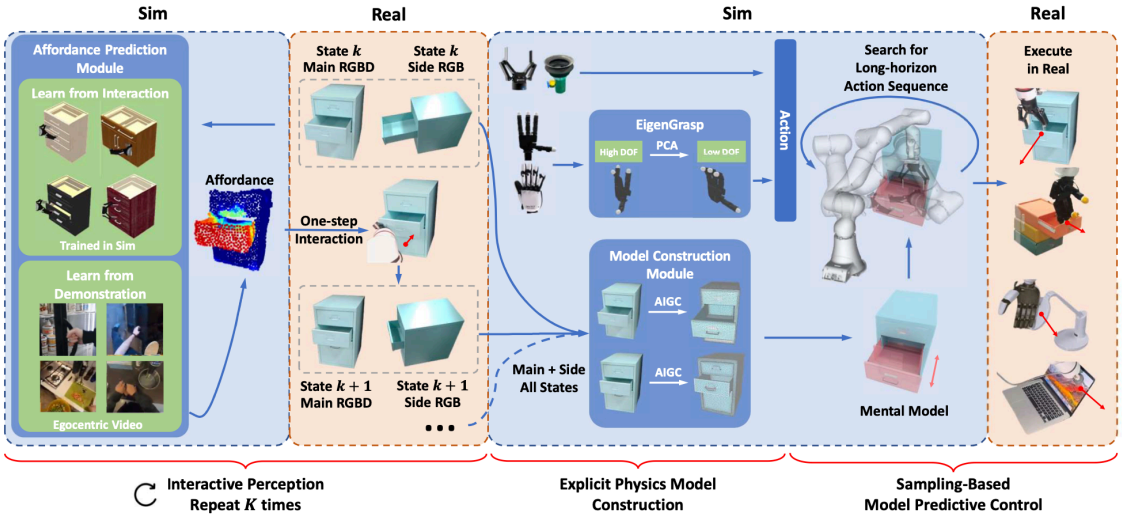

Figure 2.8: Overview of the DexSim2Real2[18] framework. The framework consists of three phases. (1) Given a partial point cloud of an unseen articulated object, in the Interactive Perception phase, they train an affordance prediction module and use it to change the object's joint state through a one-step interaction. Training data can be acquired through self-supervised interaction in simulation or from egocentric human demonstration videos. (2) In the Explicit Physics Model Construction phase, they build a mental model in a physics simulator from the K + 1 frames of observations. (3) In the Sampling-based Model Predictive Control phase, they use the model to plan a long-horizon trajectory in simulation and then execute the trajectory on the real robot to complete the task. For dexterous hands, an eigengrasp module is needed for dimensionality reduction.

However, these prior works focus exclusively on single-object modeling and thus lack the capacity to handle interactions among multiple objects. In this thesis, we propose a framework that constructs explicit world models for multiple objects and performs interactive planning for diverse tasks within a physical simulation environment for mobile manipulation.

2.5. Motion Planning for Manipulation

Obstacle-free motion planning for manipulators has been widely studied. Classical approaches rely on geometric and kinematic models to plan collision-free trajectories in the configuration space, such as RRT-connect [50] and probabilistic roadmap (PRM) [51].

While global methods (e.g., sampling-based or trajectory optimization) such as [50, 52, 53, 54], are powerful, their long computation times make them unsuitable for dynamic environments with high-DOF robots. Local planners, though with slightly lower success rates [55], enable real-time updates and adaptability, and can be categorized into geometric [56, 57], sampling-based [58, 59], and optimization-based [60, 61, 62] approaches.

Geometric methods (e.g., dynamic fabrics) are used for reactive and local motion planning without the need to solve an optimization problem. The reactive behavior of the system is described using second-order differential equations. Different components or tasks, such as collision avoidance and joint limit avoidance, are described using a second-order differential equation for each task and combined within the configuration space. Due to the absence of an optimization problem and control horizon, the computation time is fast (approx. 1 ms) which is beneficial for dynamic real-world environments. Geometric fabrics are further extended to dynamic fabrics [57] accounting for the velocities of obstacles.

Nevertheless, the primary focus of this thesis lies in the perception pipeline of the explicit world model. Therefore, we employ RRT-Connect as the motion planner for the manipulator, which provides an efficient and reliable planning solution for our experiments.

3

Methods

3.1. System Overview

Our overall objective is to develop a pipeline that constructs an explicit world model enabling action sampling, simulation, and evaluation for open-set object manipulation, conditioned on visual observations I and natural language task instructions C, as formulated in Equation 1.2. Since we leverage the existing physical simulator Isaac Sim, which already embeds rich physical priors, our focus is on constructing a digital twin that captures object-specific attributes—including geometry, material, and pose—so that it can be seamlessly integrated into the simulation for physically grounded reasoning.

An overview of the full framework is illustrated in Figure 3.1. Given an RGB-D image of the scene and a text instruction (e.g., "put the blue cup standing with its opening upside on the right wooden block"), the system outputs a complete manipulation plan and executes it on a real-world manipulator. To achieve our goal, the framework consists of four main components: **Open-set Segmentation** (Section 3.2), **Open-set Object Grasping** (Section 3.3), **Digital Twin Construction** (Section 3.4), and **Manipulation Strategy Sampling** (Section 3.5).

To accomplish a task, the robot must first detect the relevant objects in the scene. Traditional detectors are typically limited to a closed set of predefined classes and thus often fail in open-world scenarios. To address this, our **Open-set Segmentation** module leverages a combination of the vision-language foundation model GPT-4o [41] and Grounded-SAM [36] to detect and segment arbitrary objects involved in the task. Next, in the **Open-set Object Grasping** module, we generate candidate grasp poses and filter them using the segmentation masks obtained previously. After a successful grasp, the robot must reason about how to manipulate the object to achieve the task goal, which motivates the use of a digital-twin-based explicit world model. In the **Digital Twin Construction** module, we generate 3D models of the observed objects, estimate their 6-DoF pose and scale to align them with real-world observations. Besides, by predicting their materials, we also set their physical properties accordingly in the simulator. Finally, in the **Manipulation Strategy Sampling** module, we sample multiple manipulation strategies, simulate and evaluate their success probabilities. This allows us to evaluate candidate strategies within the online-constructed explicit world model and select a successful one to execute on the real robot.

3.2. Open-set Segmentation

Open-set segmentation serves as the first step of our framework, aiming to identify the objects involved in the task from an unconstrained scene without relying on a fixed object vocabulary. Given an RGB image and a natural language task prompt, our system first determines which objects in the scene are relevant to the task. For each task, the target objects are categorized into two groups: objects to grasp, which are directly manipulated (e.g., grasped and placed), and other interactive objects, which provide spatial or physical context but are not directly manipulated, such as containers, support surfaces, or target locations.

In our **Open-set Segmentation Module**, we employ GPT-4o [41] and Grounded-SAM [36] to segment

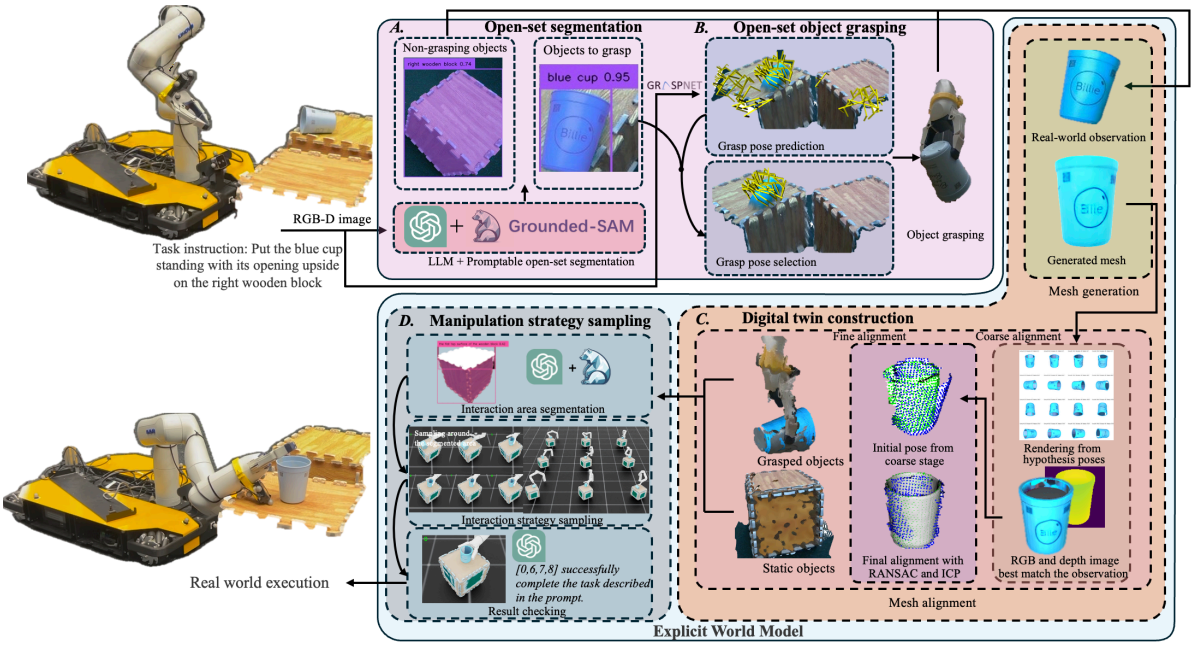

Figure 3.1: Overview of our manipulation task framework with explicit world model construction. The system takes an RGB-D observation I and a natural language task instruction C as input and proceeds through four main stages: (A) *Open-set segmentation*, (B) *Open-set object grasping*, (C) *Digital twin reconstruction*, (D) *Manipulation strategy sampling*. The outcome is the most promising manipulation strategy, which is then executed on the real robot.

any objects related to the given task prompt. Specifically, we query the GPT-4o with the RGB image and a prompt-aware question (e.g., "What objects in the picture are involved in the task 'put the blue cup standing with its opening upside on the right wooden block'? Return their names in two categories as directly manipulated objects and other interactive objects."), and the model returns two lists of candidate object names. This open-ended querying allows our method to generalize to arbitrary objects beyond a closed set of predefined classes, thanks to the scalability of GPT-4o. Once the object names are identified, we use Grounded-SAM to segment out the target objects from the input image, where the textual object names returned by the GPT-4o are used as prompts to segment object masks from the RGB image. This process results in accurate 2D segmentation masks $\tilde{M}_{\text{obj}}$ for both directly manipulated objects (e.g., cup) and other interactive objects involved in interaction (e.g., right wooden block).

$$\tilde{M}_{\text{obj}} = f_{\text{seg}}(I_{\text{rgb}}, \text{text}). \tag{3.1}$$

Despite the powerful text–image alignment capability of Grounding-DINO, the vision model still lacks spatial reasoning ability. For example, when given a segmentation prompt such as "the wooden block on the right side" for an image containing two wooden blocks, Grounding-DINO typically detects both blocks with similar confidence scores. In contrast, such spatial reasoning is a basic competency for large foundation models like GPT-4o.

To enhance the spatial reasoning ability of our perception module—which is essential for understanding spatially related tasks—we introduce an additional mask filtering stage guided by GPT-4o. Specifically, for all candidate masks returned by Grounded-SAM, we query the GPT-4o again to determine which mask best aligns with the given segmentation prompt. With this large-foundation-model-based spatial filtering, our segmentation module becomes capable of accurately identifying objects even when spatial cues are involved.

These masks are subsequently used to guide downstream modules, including instance-level grasp pose selection and digital twin construction.

3.3. Open-set Object Grasping

3.3.1. Grasp pose predictor

We adopt AnyGrasp [63] as our grasp pose prediction module. AnyGrasp is a geometry-based method trained on a large number of real-world grasping data that has demonstrated strong robustness in open-set, general-purpose grasping tasks using a parallel gripper, making it well suited for our application. The grasp pose predictor, denoted as $f_g(\cdot)$, takes an input RGB-D image I and outputs a set of grasp pose candidates $\tilde{g}$:

$$\tilde{g} = f_g(I_{\text{rgbd}}). \tag{3.2}$$

However, the predicted grasp candidates are densely distributed and correspond to all graspable objects in the scene. To reduce computational overhead, we first retain only the top 1000 predictions ranked by confidence. For instance-level grasping, this subset must be further filtered to retain only the candidates associated with the object of interest, as identified in the segmentation stage. This filtering process is carried out by our grasp pose selection module, which is described in the following subsection.

3.3.2. Grasp pose selector

The grasp pose selector is responsible for filtering the predicted grasp candidates to retain only those associated with the object of interest. To achieve this, we reuse the masks $\tilde{M}_{\text{obj}}$ of the target objects obtained from the open-set segmentation module.

Given the $\tilde{M}_{\text{obj}}$ of the target objects, we perform back-projection on the masked RGB-D image I_{obj} to obtain the corresponding partial 3D point cloud P_{obj}. Each grasp candidate in $\tilde{g}$ is then evaluated based on its spatial proximity to the surface of P_{obj}. Specifically, we retain only those grasp poses whose contact points lie within a predefined distance threshold from the segmented object's surface points,

$$s(g) = \begin{cases} 1, & \text{if } d(g) \leq \tau \quad \text{(select)} \\ 0, & \text{if } d(g) > \tau \quad \text{(discard)} \end{cases} \tag{3.3}$$

$$d(g) \triangleq \min_{\mathbf{p} \in \mathcal{P}_{\text{obj}}} \left\| \mathbf{x}(g) - \mathbf{p} \right\|_2 \tag{3.4}$$

The result of our grasp pose predictor and grasp pose selector is illustrated in Fig 3.2.

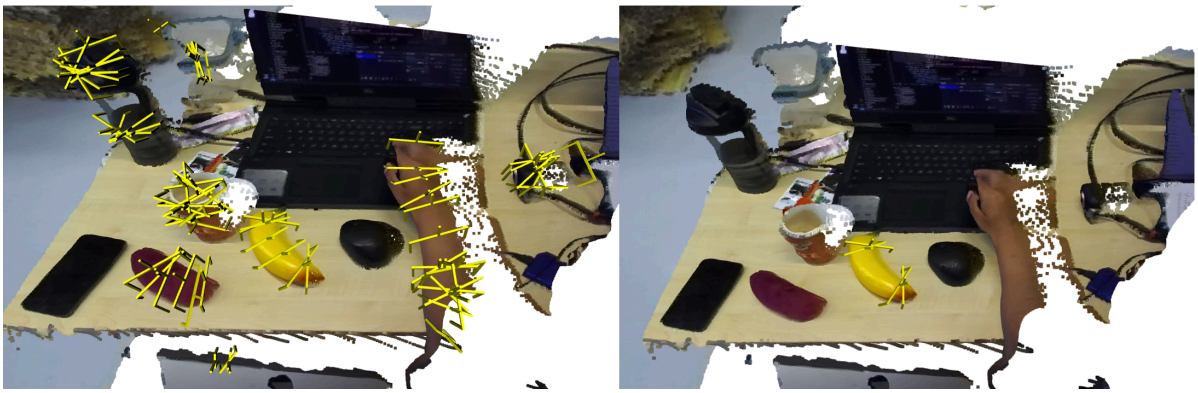

Figure 3.2: Left: all predicted grasp pose candidates generated by AnyGrasp. Right: grasp pose candidates corresponding to the banana, obtained after filtering with our grasp pose selector.

3.3.3. Grasp result checker

Similar to the large-foundation-based spatial filter in the previous module, to improve the robustness of the grasping, we also add a grasp pose checker. After the execution of grasping, we inquire the GPT-4o with the observation from the bottom camera if the gripper successfully grasped the target object,

and if the answer is no (the grasping fails or someone deliberately takes the object off), the system will keep trying to find the object and grasp it.

3.4. Digital twin Construction

Digital twin construction is one of the most critical components of our explicit world model framework. Its accuracy—such as errors in scale, pose, or material—can significantly influence the outcomes of physic-based simulation.

Therefore, the objective of this module is to construct accurate 3D meshes of the target objects, align them with real-world poses from observations, and predict their material properties, enabling their reliable use in simulation-based interaction and planning.

3.4.1. Object mesh generation

A variety of methods can be employed to construct 3D meshes. For instance, neural rendering–based approaches, such as NeRF[64] and 3D Gaussian Splatting[65], are capable of producing visually compelling 3D representations. By subsequently applying voxelization and surface extraction techniques (e.g., marching cubes), 3D meshes can be obtained from these implicit representations. However, such methods generally require densely captured multi-view images to achieve geometrically accurate reconstructions. When trained under sparse-view conditions, their performance deteriorates significantly, often resulting in incomplete or distorted geometry. Moreover, the surface quality of the meshes extracted from these representations tends to be suboptimal, exhibiting artifacts or uneven topology.

In contrast, recent 3D generation methods can produce high-quality meshes from sparse views, or even from a single image, making them particularly suitable for our objective of explicit world model construction, where both geometric accuracy and generalization from limited observations are essential. Previous works such as Xu et al.[66] and Pan et al.[67] employed DeepSDF [49] to generate complete object shapes. However, DeepSDF relies on a category-level latent code to guide generation, which must either be obtained during training or through online optimization—which is computationally expensive. This reliance hinders open-set object interaction, which is central to our task.

To overcome these limitations, this project adopts Hunyuan 3D 2.0 [68] as the 3D generation module. Hunyuan 3D 2.0 is capable of generating both shape and texture directly from a single RGB image. By offering a favorable balance between inference speed, reconstruction quality, and generalization across diverse object categories, it provides a practical and effective foundation for building explicit world models in open-world robotic manipulation.

The quality of 3D generation is also highly dependent on the input image, which should ideally capture the most distinctive features of the object. For instance, a top view of a mug often lacks informative texture cues, making it difficult for the model to even recognize it as a mug; in such cases, a front view is typically more informative. Conversely, for objects like a banana lying flat on the ground, the top view generally provides a clearer representation than the front view. Based on this prior, our robot adopts a exploratory observation strategy. It observes the objects from both front and top views, performs segmentation and bounding box cropping, and selects the view that offers a more informative perspective as input to the 3D generation module, as shown in Fig.3.3

3.4.2. Object mesh alignment

The mesh generated by the 3D generation module is in unit scale and canonical pose, meaning it captures only the object's geometry, but lacks global scale and pose information in the world coordinate frame. Therefore, it is necessary to align the generated mesh with the real-world observation by estimating the appropriate scale and 6-DoF pose.

However, this alignment is inherently challenging. Traditional point cloud registration methods—such as RANSAC and ICP (Iterative Closest Point)—typically rely on the assumption that dense and reliable point-to-point correspondences exist between the source and target point clouds. This assumption does not hold in our case: the generated mesh provides a complete and idealized geometry of the object, while the real-world observation captures only a partial and potentially noisy view, resulting in ambiguous correspondences. For example, when only a portion of a mug is observed—excluding the handle—it becomes difficult to determine the correct matching region on the complete mesh, as multiple

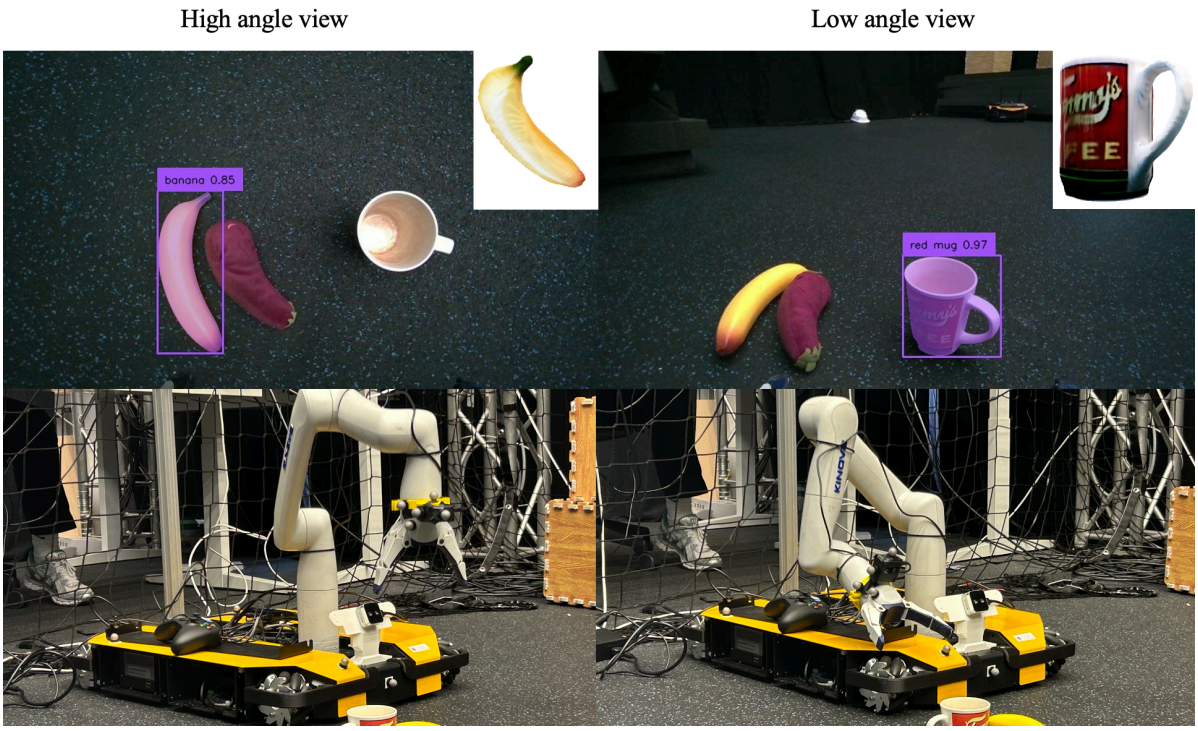

Figure 3.3: High-angle and low-angle views of real-world observations. For certain objects such as the mug, the front view provides more informative visual features and is therefore preferred for 3D mesh generation. In contrast, for objects like the banana, the top view captures more discriminative geometry, making it a better choice. The generated meshes are on the top right of each view and below are the corresponding robot poses.

locations may appear geometrically similar from that limited viewpoint. To address this, we incorporate object texture into the alignment process. Our proposed 7-DoF pose alignment pipeline (3 translations, 3 rotations, and 1 uniform scale) performs mesh-to-observation matching in a coarse-to-fine, two-stage manner, as shown in Fig 3.4.

Coarse alignment

Let $\tilde{P}$ denote the set of hypothesized coarse object poses sampled in 6D space (translation and rotation). For each hypothesized pose $\tilde{p} \in \tilde{P}$, the corresponding textured 3D mesh is rendered to obtain both the RGB image $\tilde{I}$rgb and the depth image $\tilde{I}$depth. The following equation describes this rendering process:

$$\tilde{I}_{\text{rgb}}, \tilde{I}_{\text{depth}} = \text{Render}_{\tilde{P}}(\text{Mesh}). \tag{3.5}$$

Then, we compare the similarities of these rendered views with the real-world observation I_{obs} to select the most promising hypothesis as an initial coarse pose estimate. Here we use DINOv2 [69](distillation with no labels), a very powerful SOTA image feature extractor, to transform the images from pixel space to feature space and compute the cosine similarities,

$$\tilde{F}_{\text{render}} = \text{DINO}(\tilde{I}_{\text{rgb}}) \tag{3.6}$$

$$F_{\text{obs}} = \text{DINO}(I_{\text{obs}}) \tag{3.7}$$

$$\tilde{S} = cosine(\tilde{F}_{\text{render}}, F_{\text{obs}}), \tag{3.8}$$

We then select the hypothesized pose with the highest similarity to the real-world observation as the coarse pose estimate. Using this pose, we render a depth image of the generated mesh and back-project it to obtain a partial point cloud that corresponds to the observed view. This transformation effectively converts the challenging partial-to-complete alignment problem into a partial-to-partial alignment problem, thereby enabling the use of conventional point cloud registration methods, such as RANSAC and ICP, for further fine alignment.

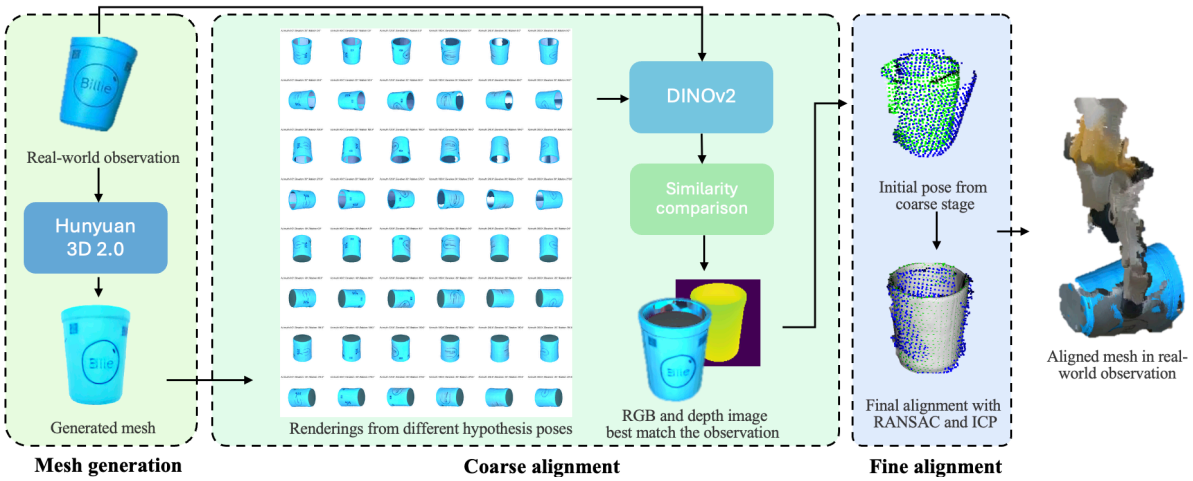

Figure 3.4: The proposed digital twin construction module, containing the mesh generation and two-stage pose alignment. We first generate a textured mesh from the masked RGB image via Hunyuan3D 2.0 [68]. During coarse alignment, we render RGB and depth images from a set of hypothesis poses and compare their similarities with real-world observation in DINO [69] feature space, and select the one that best matches the real-world observation. The resulting coarse pose is then refined using RANSAC and ICP on the partial point cloud back-projected from the depth image.

It is worth noting that for the real-world observation of other interactive objects, we can directly reuse the segmentation result obtained from the Open-set Segmentation module. However, for the grasped object, we require its observation after the grasp is completed, since we need to know the transformation between it and the gripper in the next stage. This requirement introduces a new challenge: segmentation performance on bottom-view images is often poor. We hypothesize that this is due to the unfamiliar view angle and the presence of the gripper, which are rarely seen by the pretrained detector. However, since the gripper pose is relatively fixed to the bottom camera after grasping, we can reliably segment the grasped object using Segment Anything (SAM) [37], by providing bounding boxes and point prompts as inputs, as illustrated in Fig. 3.5.

Fine alignment

The coarse alignment produces two roughly aligned point clouds that still differ in scale. To estimate the scale factor, we compare the dimensions of the 3D bounding boxes derived from the partial point clouds of both the rendered mesh and the real observation, and adjust the mesh dimensions to match the real-world object. Following this, we refine the transformation using a combination of RANSAC and ICP, yielding an accurate alignment between the generated mesh and the real object. We examined our mesh alignment pipeline and the results are in Sec. 4.3

3.4.3. Material property prediction

Inspired by Xu et al. [70], who leveraged a large language model (LLM) together with a curated material library to infer the material and physical properties of an object from its masked image, we adopt a similar idea by utilizing the reasoning capability of GPT-4o to predict the material of the object. The predicted material can then be used to assign corresponding physical properties in the simulation environment.

3.5. Manipulation Strategy Sampling in Simulation

Once the digital twin of the environment is constructed, we integrate it into a physics-enabled simulation environment to perform manipulation strategy sampling. In this work, we adopt NVIDIA Isaac Sim as the simulation platform. We then sample 6-DoF poses (three translations and three rotations) for the dynamic object, representing candidate manipulation goals. Each candidate pose is instantiated in the simulated scene, and the resulting physical interaction is simulated using the physics engine. The outcome of each simulation is then evaluated to determine whether the manipulation strategy successfully completes the task. This process allows us to test diverse candidate strategies without executing them on the real robot.

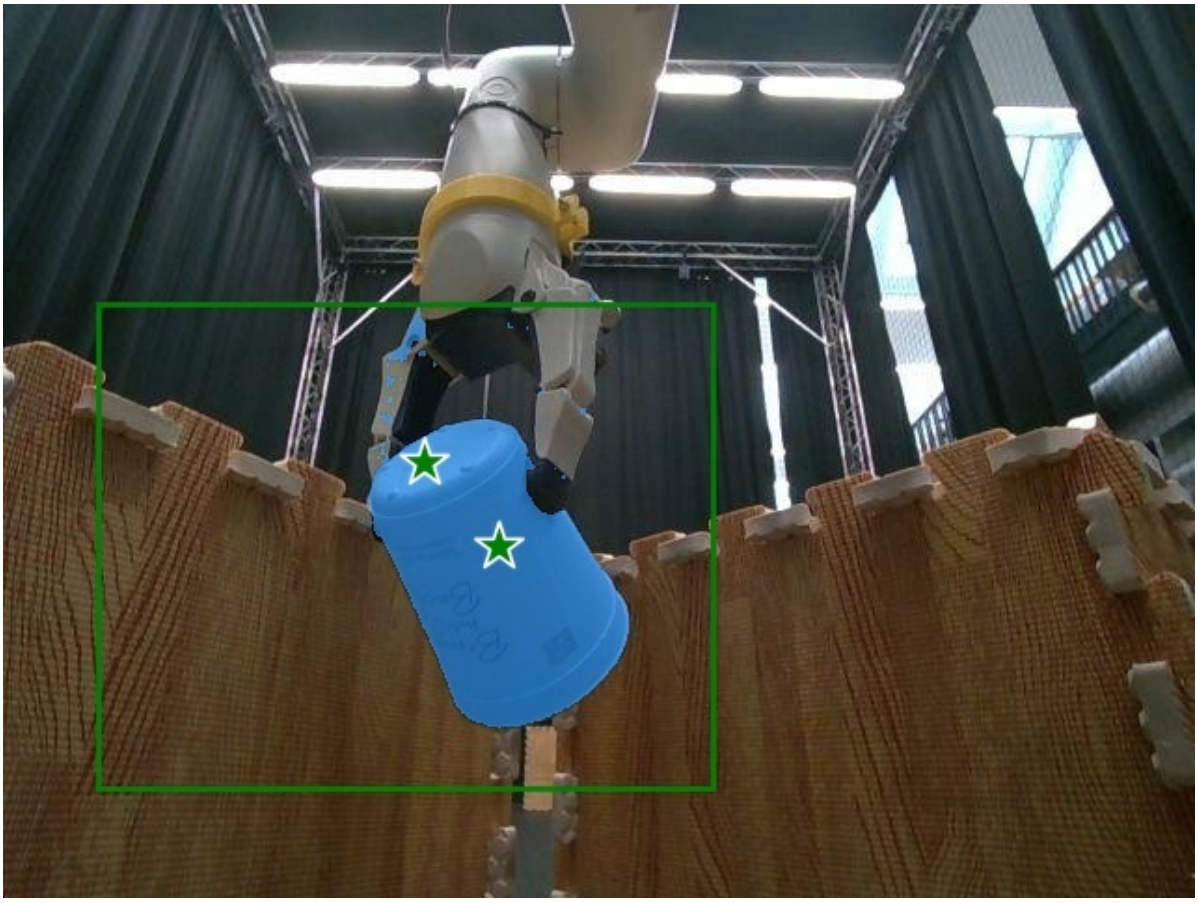

Figure 3.5: The segmentation result on the bottom-view image, with points and the box prompts annotated. The two green points indicate the foreground area.

3.5.1. Interaction area segmentation

To improve sampling efficiency and reduce the size of the search space, the translation component of the dynamic object's pose is constrained to be near the most probable interaction region, as illustrated in Fig. 3.6. Specifically, we use the masked image of the non-directly manipulated object to query GPT-4o about which part of the object the dynamic one is expected to interact with. The identified region is then segmented using Grounded-SAM, which provides a precise mask of the interaction area. The segmented region is back-projected into 3D space to obtain a point cloud, from which we compute the centroid as reference for the initial estimate of the translation component.

3.5.2. Interaction strategy sampling

After constraining the translation around the interaction area, we need to sample different rotation angles to generate diverse 6-DoF pose hypotheses. For each sample, we spawn the non-directly manipulated object at its real-world pose and the grasped object at the sampled pose. Accordingly, we also need to spawn the Kinova arm with proper joint values, since there is a constant offset between the gripper's tool frame and the center of the grasped object. Specifically, for each grasped object, we have an offset vector t_{to},

$$p_{toolframe} = p_{obj} + t_{to} \tag{3.9}$$

where $p_{toolframe}$ is the gripper tool frame position and p_{obj} is the object position. We also spawn the gripper in open state, because otherwise it will be in collision with grasped object at spawning, leading to abnormal collision behavior.

Since there's a connection between the grasped object pose and the robot arm end effector pose, we need to further constrain the sample space according to the robot arm's workspace. To characterize the workspace constraints of the robot arm, we compute a reachability map using MoveIt!. Specifically,

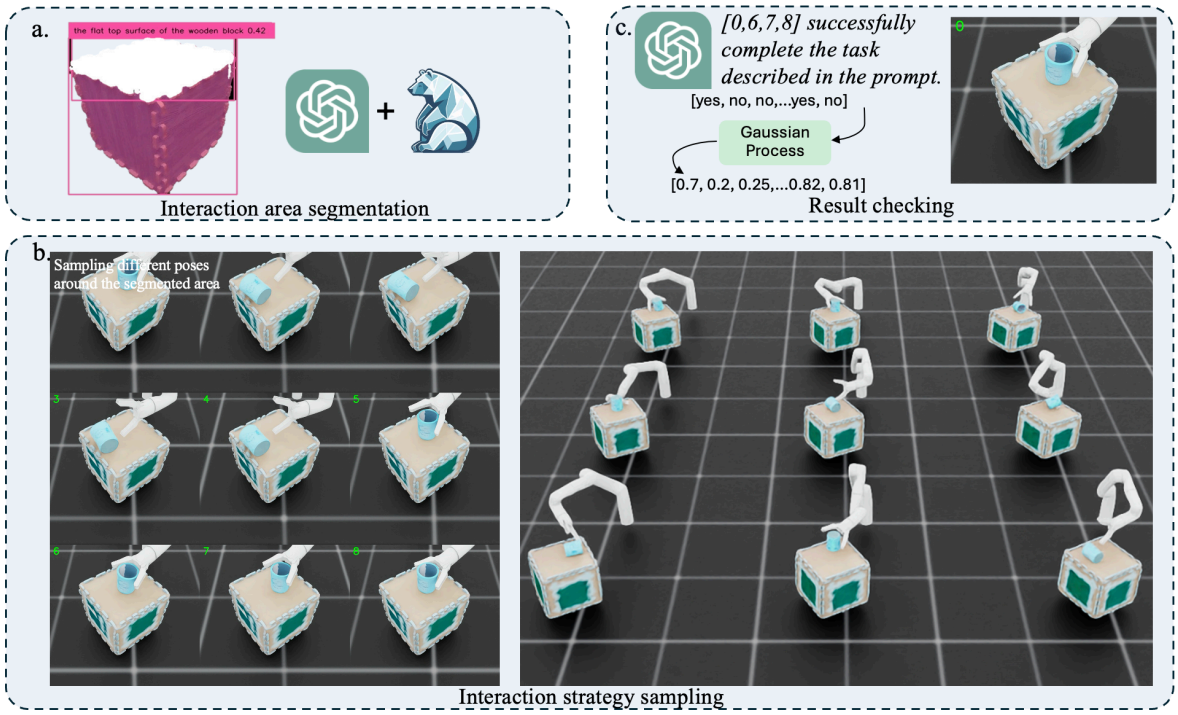

Figure 3.6: Our manipulation strategy sampling module. (a) *Interaction area segmentation*: We use GPT-4o and Grounded-SAM to segment the interaction area on the not directly manipulated interactive object to constrain the sample space of translation. (b) *Interaction strategy sampling*: Different rotations of the dynamic object are sampled and the interactions are simulated in the Isaac Sim. (c) *Result checking*: The results are rendered in the simulator. We query the GPT-4o again to check which samples fulfil the task requirements, and feed the results to a Gaussian Process classifier to get the success probabilities. The candidate with the highest success probability will be executed on the real robot.

8,000 samples are generated by randomly assigning joint values to the arm. For each sample, the corresponding end-effector pose is recorded and assigned to its nearest voxel and rotation bin. Given that our task requires fine-grained manipulation, the voxel size is set to 1 cm, and the orientation space is discretized into 720 rotation bins, uniformly sampled on the sphere. Although uniform sampling in the 3D Euler angle space would be simpler, it leads to severe non-uniformity; therefore, spherical sampling is adopted instead.

The resulting reachability map is stored as a static library. For any queried end-effector position, we first determine the voxel it belongs to and check whether this voxel exists in the reachability map. If it does, we directly retrieve its associated rotation samples from the library. Otherwise, we search for its nearest neighboring voxel within a predefined range. To further improve coverage and exploration, we apply farthest point sampling to the retrieved rotation candidates and introduce small angular perturbations to diversify the sampled orientations.

For each sampled reachable end-effector pose, we compute the corresponding joint configuration using the MoveIt! inverse kinematics (IK) solver. Since the RRTConnect planner produces randomly sampled suboptimal trajectories, we execute the planner ten times for each end-effector pose and select the trajectory with the smallest joint-space distance. After obtaining all joint configurations, we spawn the robot arm at these joint positions in the simulator, as illustrated in Fig. 3.6b (right).

3.5.3. Result checking and action selection

As formulated in Equation 1.2, we still require a score function $R(C, \tau)$ to evaluate the success of a candidate trajectory $\tau = (s, o)$, where s and o denote the future states and observations generated by the world model W, respectively. Since the internal states s are not directly needed for evaluating task outcomes in our formulation, the function can be simplified to depend only on the visual observations, i.e., $R(C, o)$. Specifically, we use a combination of a large-foundation-model-based visual-language identifier with a Gaussian Process (GP)–based success probability estimator to approximate the score

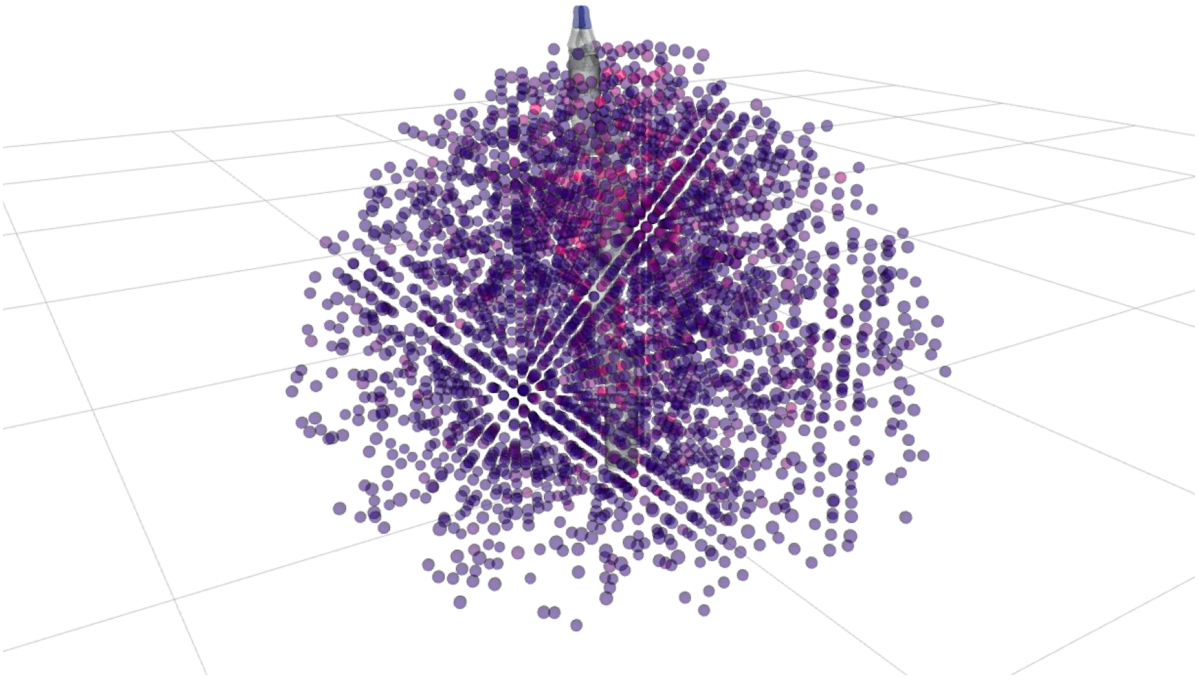

Figure 3.7: RViz visualization of the computed reachability map. The color encodes the sampling density: blue indicates regions with fewer reachable samples, while red highlights regions with higher sampling density.

function. The former assesses semantic task completion by comparing rendered observations under the given instruction C, while the latter provides a probabilistic estimate given the binary success/failure label from the former.

For each simulated strategy sample, we render an RGB image as the observation for each candidate from a fixed, front-facing viewpoint with a $-60°$ tilt, shown in Fig 3.6 b on the left. We then query GPT-4o with the rendered image and the natural language task instruction to determine whether the observed outcome satisfies the instruction. This yields weak binary labels $y_i \in \{0, 1\}$ for each end-effector pose $(\mathbf{t}_i, \mathbf{q}_i)$, where $\mathbf{t}_i \in \mathbb{R}^3$ is translation and $\mathbf{q}_i = [w\ x\ y\ z]^\top$ is a unit quaternion. The strategies deemed successful by the GPT-4o are prioritized for real-robot execution.

To obtain calibrated success probabilities, we train a Gaussian Process (GP) classifier over SE(3). We map each pose to a 6D Euclidean feature by combining translation and rotation with explicit scaling:

$$\boldsymbol{\phi}(\mathbf{t}, \mathbf{q}) = \left[\ (\alpha_t/\ell_t)\,\mathbf{t}^\top,\ \ (1/\ell_r)\,\mathbf{r}(\mathbf{q})^\top\right]^\top \in \mathbb{R}^6, \tag{3.10}$$

where $\mathbf{r}(\mathbf{q}) \in \mathbb{R}^3$ is the axis–angle rotation vector obtained from the unit quaternion with antipodal identification (we enforce $w \geq 0$ so that $\mathbf{q}$ and $-\mathbf{q}$ map to the same rotation). The hyperparameters ℓ_t and ℓ_r control the sensitivity to translation and rotation, and $\alpha_t \in (0, 1]$ further down-weights translation relative to rotation (The rotation is more important in sampling, as we already have a prior knowledge for translation via interaction area segmentation). In practice, we set ℓ_t=1.0m, ℓ_r=10° (in radians), and α_t=0.2.

On this feature space we use an RBF kernel with a learnable length scale,

$$k(\boldsymbol{\phi}, \boldsymbol{\phi}) = \sigma^2 \exp\Big(-\tfrac{1}{2\ell^2}\,\|\boldsymbol{\phi} - \boldsymbol{\phi}\|_2^2\Big), \tag{3.11}$$

and fit a Gaussian Process classifier (Laplace approximation) on the LLM-provided labels $\{(\mathbf{t}_i, \mathbf{q}i), y_i\}$. We perform a small grid search with stratified K-fold cross-validation to choose a good initial ℓ for the optimizer. At test time, the GP outputs calibrated success probabilities,

$$p_i \ = \ \Pr\big(y{=}1 \,\big|\, \mathbf{t}_i, \mathbf{q}_i\big), \tag{3.12}$$

Algorithm 1: Reachability Map Construction and Query

Input: Robot arm model, voxel size $v = 1$cm, number of rotation bins $N_r = 720$
Output: Static reachability map $\mathcal{R}$

Construction phase:
for $i \leftarrow 1$ **to** 8000 **do**
    Randomly sample joint values q_i;
    Compute end-effector pose T_i from q_i;
    Assign position of T_i to nearest voxel V_j;
    Assign orientation of T_i to nearest rotation bin R_k;
    Store (V_j, R_k) in $\mathcal{R}$;

Query phase:
Use object target position to approximate end-effector position p;
Find voxel V_p containing p;
if $V_p \in \mathcal{R}$ **then**
    Retrieve associated rotation samples $R(V_p)$;
else
    Search nearest neighboring voxel V_{nn} in range;
    Retrieve rotation samples $R(V_{nn})$;
Apply farthest point sampling to $R(\cdot)$;
Add small angular perturbations to enhance coverage;
return feasible set of rotation candidates;

We use the predicted probabilities to rank all candidate strategies and select the one with the highest likelihood of success for real-robot execution. However, we do not directly optimize the trained Gaussian Process model to search for the pose $(\mathbf{t}_i, \mathbf{q}_i)$ with the maximum predicted probability, as this does not guarantee physical feasibility or reachability within the robot's workspace.

4

Experiments

This chapter details the experiments we designed to validate and benchmark the proposed solution. The experiments validate the performance of several main components in our system: the accuracy of digital twin reconstruction, the success rate of the strategy sampling in both simulation and real robot execution.

4.1. Hardware Setup

The robot setup is shown in Fig 4.1. We used the 7-DoF Kinova Gen3 Lite robotic arm [1]. Two Intel RealSense D405 stereo cameras [2] were mounted on the gripper and the front bottom on the mobile base with 3D printed camera supports respectively, serving as the primary and auxiliary perception sensors. The bottom camera also has a 25 degree elevation angle to better observe the grasped object. For computational resources, we have a NVIDIA RTX 2070 GPU with 6GB RAM on a laptop and a NVIDIA RTX 3090 GPU workstation with 24GB RAM.

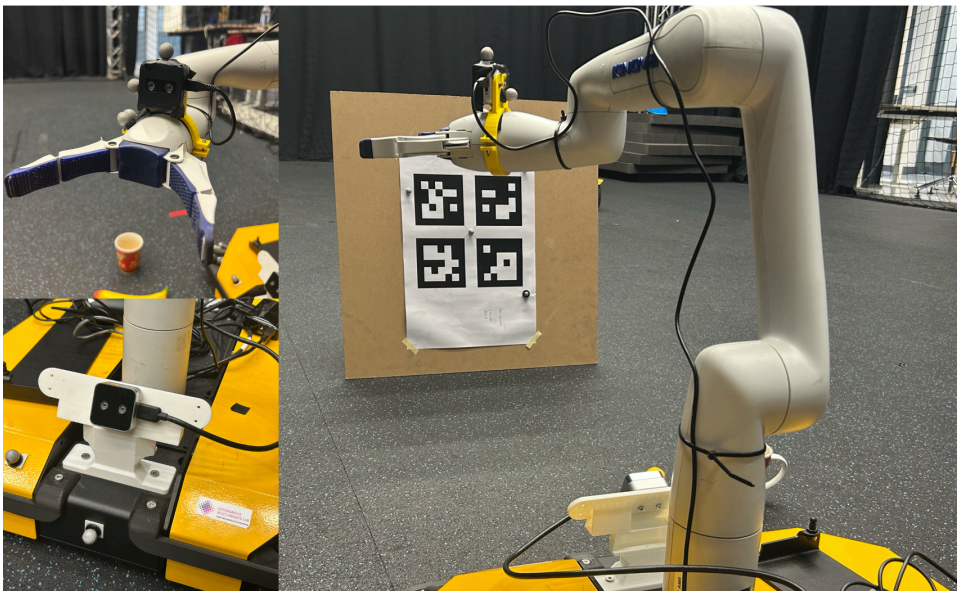

Figure 4.1: The sensor setup of our system. We mounted two RealSense D405 stereo cameras on the gripper (top left) and the front bottom on the mobile base (top right). For the gripper camera, we use Vicon system to get its pose in world frame, and for the bottom camera, we use calibration board to get the fix transformation between the mobile base and itself.

To obtain accurate pose information of both the Kinova base link and the gripper-mounted camera, we

[1] https://www.kinovarobotics.com/product/gen3-lite-robots
[2] https://realsenseai.com/stereo-depth-cameras/stereo-depth-camera-d405/

employed a Vicon motion capture system for external tracking. The transformation between the bottom camera and the robot was acquired from calibration.

4.2. Software Setup

The software architecture of the system is illustrated in Fig. 4.2. To handle computationally demanding tasks, GPU-intensive functional modules—including AnyGrasp, Grounded-SAM, and Hunyuan 3D—are deployed on a remote server equipped with an NVIDIA RTX 3090 GPU. The main control laptop connects to the remote server via SSH, transmitting data and retrieving results through a Flask-based communication service. Within the local network, the user can issue text prompts to the main control laptop to initiate task execution, while the robot communicates with the laptop through the ROS framework for data exchange and control. The strategy sampling is also performed on the main control laptop, with Isaac Sim 4.5.0 + Isaac Lab 2.2.1.

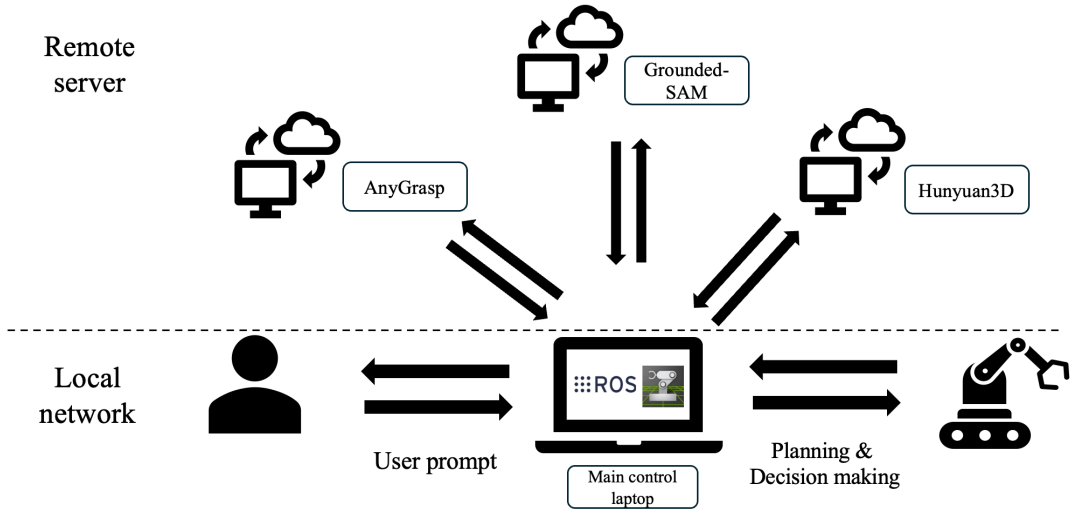

Figure 4.2: The software setup of our system. Within the local network, the user provides commands to the main control laptop, which communicates with the robot via ROS to fetch sensor data and send control signals. The laptop also interfaces with remote servers running AnyGrasp, Grounded-SAM, and Hunyuan3D for grasp detection, segmentation, and 3D shape generation. The Isaac Sim simulation sampling is also run on the main control laptop.

4.3. Digital Twin Construction Accuracy

The objective of our digital twin reconstruction module is to obtain an aligned mesh of the target object that accurately restores its geometry and global pose. To the best of our knowledge, our approach is novel in its open-set property—it requires no category-specific prior knowledge, taking only a single RGB-D image as input.

In this experiment, we compare our proposed two-stage mesh alignment pipeline with a direct alignment baseline, which applies RANSAC and ICP directly to the generated mesh. Ideally, pose error would serve as a more informative metric for evaluating alignment performance; however, ground-truth object poses are unavailable in our real-world setup, and coordinate frames are not consistently fixed for each generated mesh. Therefore, we evaluate alignment performance using two complementary metrics: alignment success rate, which measures the correctness of the estimated object pose, and root mean square error (RMSE), which assesses the geometric precision of the alignment.

We conduct experiments across multiple object categories, both being placed freely and under grasping scenarios. Since ground-truth meshes are also unavailable, we use Hunyuan 3D to generate meshes for all samples. Both the proposed and baseline alignment pipelines use the same generated meshes for a fair comparison. The quantitative results are summarized in Table 4.1.

The results demonstrate that our proposed two-stage mesh alignment method substantially improves the alignment success rate compared with direct alignment using RANSAC and ICP. The direct alignment baseline relies solely on minimizing point-to-point distances without a reliable initialization, which

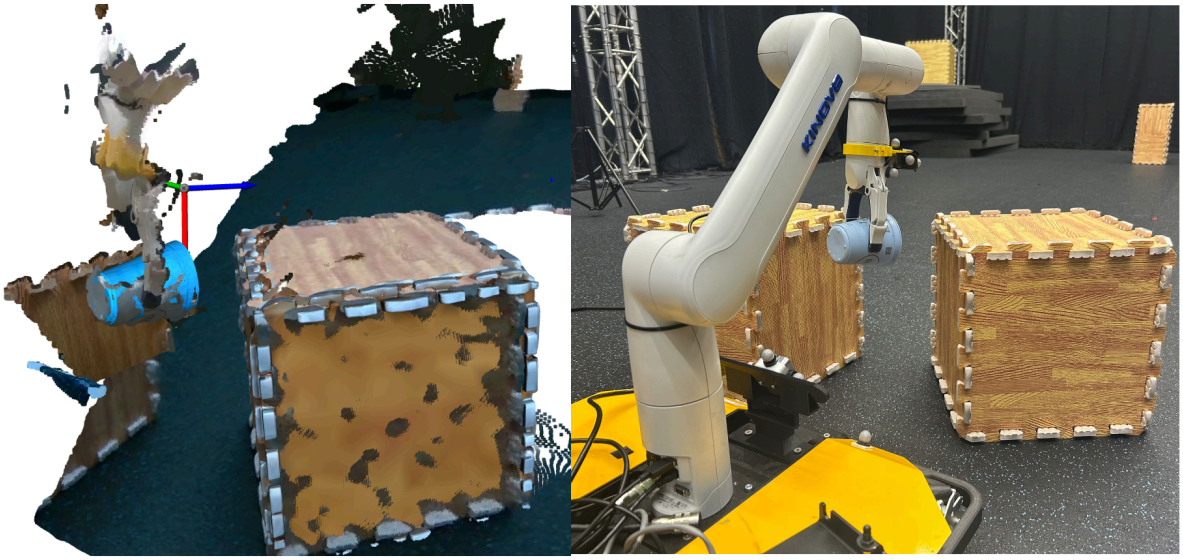

Figure 4.3: Qualitative results of the proposed two-stage mesh alignment method. The left image shows the generated meshes of the blue cup and wooden block aligned and overlaid with the RGB-D observation. The right image illustrates the corresponding real-world setup.

objects	valid sample	our method		direct alignment	
		success rate ↑	RMSE ↓	success rate	RMSE
without grasp in the gripper					
mug	11	**90.91%**	0.00457	27.27%	**0.00390**
box	5	**100.00%**	**0.00564**	**100.00%**	0.00684
foam box container	5	**40.00%**	**0.00665**	20.00%	0.00745
bottle	8	**100.00%**	0.00432	62.50%	**0.00367**
laptop	11	**90.91%**	**0.00408**	45.45%	0.00453
grasped in the gripper					
mug	9	**88.89%**	**0.00443**	11.11%	0.00450
banana	8	**100.00%**	0.00505	**100.00%**	**0.00460**
fish	4	**100.00%**	0.00400	**100.00%**	**0.00317**
cup	3	**66.67%**	**0.00412**	33.33%	0.00474

Table 4.1: Comparison of mesh alignment performance between our two-stage alignment pipeline and direct alignment under different conditions (with and without grasped objects). The table reports the alignment success rate and root mean square error (RMSE) for each object category.

often leads to convergence to local minima or incorrect correspondences—particularly in cases where the partial point cloud differs significantly from the complete mesh. In contrast, our method leverages an initial coarse alignment stage guided by appearance similarity, effectively narrowing the search space for fine alignment and improving robustness to noise and scale differences.

Although in a few cases the direct alignment yields slightly lower RMSE values, these typically correspond to the limited instances where it converges correctly. Overall, our method achieves a much higher alignment success rate while maintaining comparable geometric accuracy, validating the effectiveness of the two-stage pipeline for robust and generalizable mesh-to-scene alignment.

4.4. Real Robot Task Performance

4.4.1. Experiment setup

We performed three real robot experiments for a total 72 attempts on our system, with 24 attempts for each. The selected tasks are designed to validate the system's ability to understand both semantic instructions and spatial relations, and also the ability to complete various open-world tasks without seeing the objects before or fine-tuning on these tasks. Relying on these properties, our system shows

the potential to tackle the open-world object manipulation problem in a novel way. The tasks and their corresponding text prompts are illustrated in Fig 4.4. Fig 4.5 shows all the objects we used in the experiments.

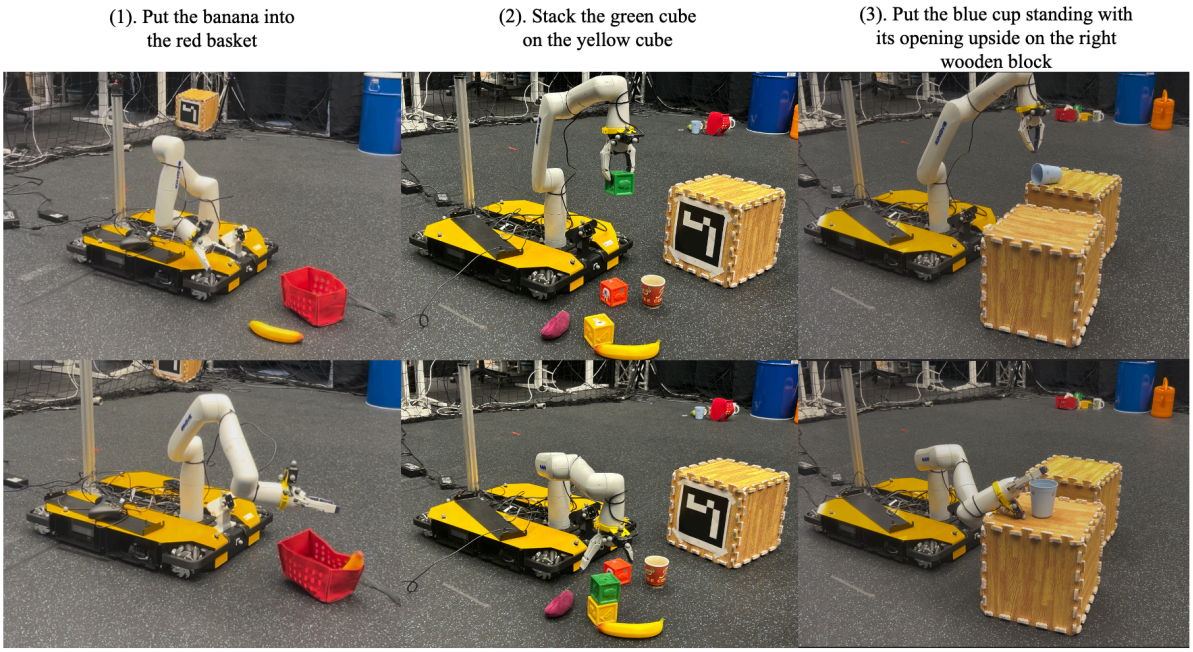

Figure 4.4: Three designed tasks for the validation of our explicit world model system.

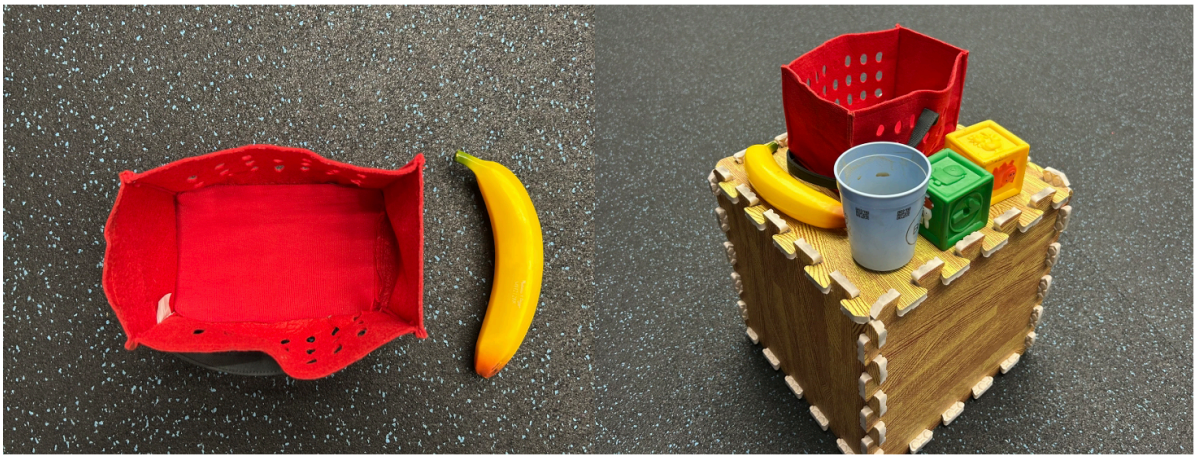

Figure 4.5: The relative scales of the objects used in our manipulation tasks.

The system parameters used in all experiments are summarized in Table 4.2. For each trial, the number of strategy samples simulated in Isaac Sim is fixed to nine. This number controls the trade-off between computational efficiency and robustness: a smaller number accelerates the overall simulation but increases the risk of not obtaining any successful sample, whereas a larger number improves the chance of finding feasible strategies at the cost of longer simulation time The dropping margin refers to the distance along the z-axis between the centroid of the interaction area and the sampled dropping position. A larger margin helps prevent collisions between the grasped object and nearby objects, whereas a smaller margin can be advantageous in tasks where object bouncing is undesirable. Additionally, the RRT-Connect planner is configured with two key parameters: position tolerance and orientation tolerance. Smaller tolerances result in more precise trajectory execution but increase the likelihood of planning failure.

Parameter	Value	Description
Number of samples	9	number of samples used in simulation
Dropping margin	0.6 * maximum dimension of the grasped object	the z-axis distance between the centroid of the interaction area and the sample position
RRTConnect planner position tolerance	0.03m	the maximum error of the tool frame position allowed in the planning
RRTConnect planner orientation tolerance	0.08rad	the maximum error of the tool frame orientation allowed in the planning

Table 4.2: System parameters used in the experiment.

4.4.2. Quantitative results

The respective success rate of three tasks are listed in Table 4.3. From the statistics, we can see that ”Put the banana into the basket” is the ”easiest” task, as it has a relatively high success rate, while the other two tasks share similar low success rates.

Task	Success Rate
Put the banana into the basket	62.5%
Stack the green cube onto the yellow cube	16.7%
Put the blue cup upside down on the wooden box	12.5%

Table 4.3: Task success rate of different manipulation scenarios.

Overall results

To provide a comprehensive evaluation of the entire system, we record the success/failure results of all major components, including object grasping, mesh generation, mesh alignment, interaction area segmentation, strategy sampling in simulation, sampling result checking, and real robot execution. The results are visualized as a Sankey diagram as shown in Fig 4.6. The definitions of each failure type are detailed in Table 4.4.

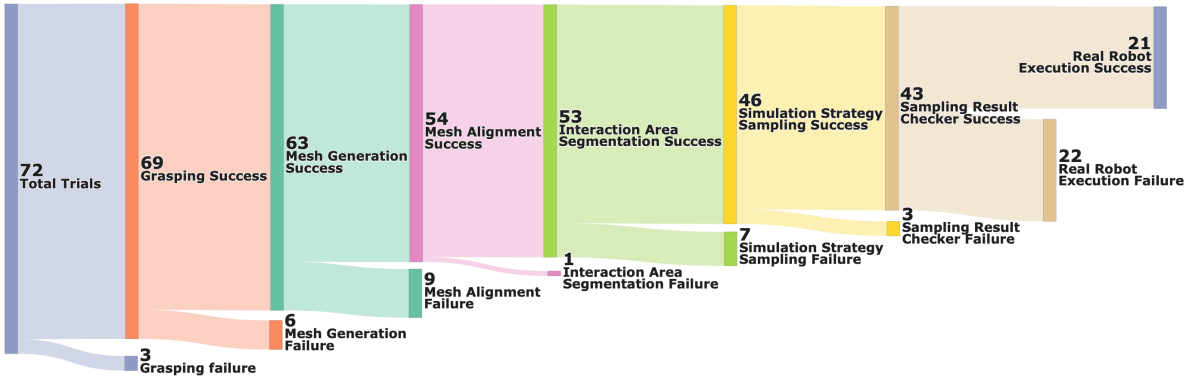

Figure 4.6: Sankey diagram illustrating the stage-wise success and failure flow across all 72 real-world trials. Each block represents one functional module in the proposed framework, while the link thickness corresponds to the number of trials that succeeded or failed at that stage. The results show that most failures occur during mesh alignment and real-robot execution, indicating that perception accuracy and physical execution robustness are the major bottlenecks of the current system.

Figure 4.6 illustrates the stage-wise success and failure flow across all 72 real-world trials. The early stages of the framework, including grasping, mesh generation, and mesh alignment, achieve relatively high success rates (the third task contributes 8 out of 10 mesh alignment failure cases), indicating that the perception and digital twin reconstruction modules are stable and robust under various conditions. In contrast, the final stage — real robot execution — shows a noticeably lower success rate, with only 21 successful completions out of 42 valid strategies verified by the checker.

Failure Type	Definition
Grasping Failure	The robot failed to grasp the object within three attempts. Since a grasp result checker is included, the robot will theoretically keep retrying until a successful grasp is detected.
Mesh Generation Failure	The generated mesh exhibits incorrect geometry or major structural differences compared with the real object.
Mesh Alignment Failure	The estimated mesh pose significantly deviates from the true pose of the real object. Minor deviations within the acceptable tolerance range are not considered as failures.
Interaction Area Segmentation Failure	The segmented interaction region is clearly incorrect, e.g., in the task *"put the banana into the red basket"*, the segmented area corresponds to the basket's side instead of its opening.
Simulation Strategy Sampling Failure	No successful strategy sample was found in the simulation among the generated candidates.
Sampling Result Checker Failure	Although at least one successful candidate exists, the checker module fails to return the correct successful index.
Real Robot Execution Failure	Given a successful simulation strategy, the real robot fails to physically execute and complete the task.

Table 4.4: Definitions of the different failure types for different stages in the proposed framework.

The substantial drop in success rate in the final execution stage highlights the sim-to-real gap between simulation and the physical world: although our framework found the successful action candidates, the real robot execution might have a different result. Several factors contribute to this discrepancy. First, the physics engine of Isaac Sim, though highly realistic, cannot perfectly replicate real-world contact dynamics such as friction, compliance, and collision restitution, leading to discrepancies in manipulation outcomes. In an attempt to mitigate this gap, we incorporated material property estimation to better parameterize physical attributes (e.g., friction and restitution coefficients) in Isaac Sim. However, the range of adjustable parameters remains limited, and the simulated behaviors still deviate from real-world dynamics. In particular, hollow or deformable structures, such as plastic cubes, cannot yet be accurately modeled or detected based solely on visual appearance, leading to further inconsistencies between simulated and real interactions. Second, the robot hardware precision and control latency introduce minor pose and timing errors that can accumulate during execution, especially for tasks with tight spatial constraints. Third, sensor noise and calibration errors (e.g., camera extrinsics and depth inaccuracies) can cause small misalignments between the reconstructed scene and the actual setup, which are tolerable in simulation but critical for real-world success.

Task-specific results

In addition to the overall framework statistics presented in Fig.4.6, we conducted a task-level robustness analysis to examine how different types of failures are distributed across various object manipulation tasks. The corresponding failure distributions are illustrated in Fig.4.7.

To better interpret the experimental statistics, we first analyze the properties of each task and the characteristics of the involved objects. For the first task, "put the banana into the red basket", both the banana and the basket are relatively large in scale, which makes the task outcome more robust to minor position errors. The task also exhibits a higher tolerance to orientation inaccuracies. Moreover, since the basket is soft, it provides additional robustness to small deviations in the dropping margin — the basket is less likely to be displaced even if the object makes slight contact with it. All these factors contribute to the overall high success rate of this task. However, despite the high completion rate, this task also shows the highest number of grasping failures. The banana is a thin and rigid object, making it sensitive to vertical positioning errors of the tool center point (TCP). A slightly higher TCP causes the gripper to collide with the ground, while a lower TCP often leads to missing the object entirely.

For the second task, "stack the green cube on the yellow cube", both position and orientation tolerances

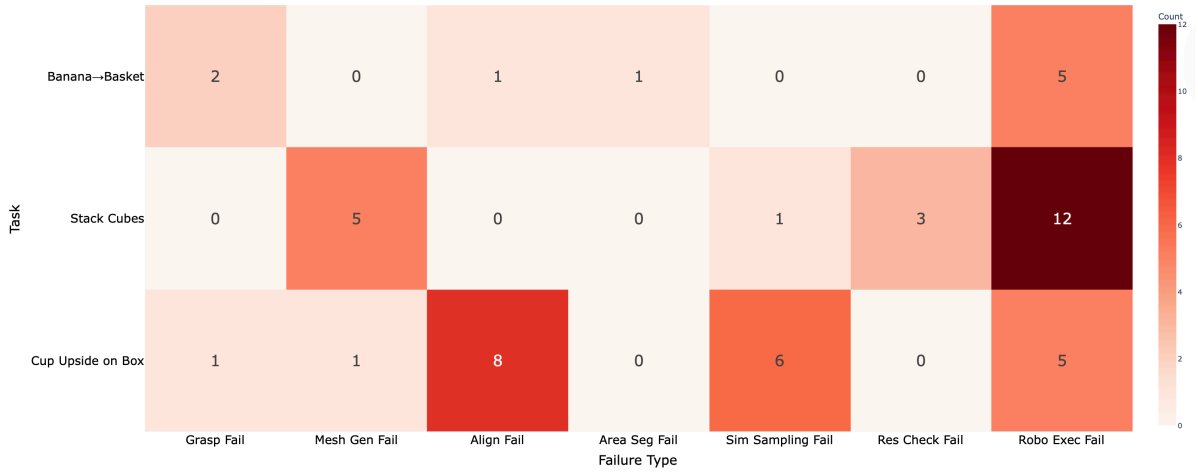

Figure 4.7: Failure type distribution across the three object manipulation tasks. Each cell represents the number of occurrences of a specific failure type for a given task. The color intensity reflects the relative failure frequency, showing that the stacking cubes task fails most frequently during the real-robot execution stage, while the cup upside on box task suffers primarily from mesh alignment errors.

are much tighter than in the previous task. The cubes are made of soft rubber and are hollow inside, which makes the interaction highly sensitive to the dropping margin. Even with accurate positioning and orientation, the top cube can easily bounce off if the dropping margin is too large, while an overly small margin may cause the bottom cube to be pushed away. Although we can adjust the restitution coefficient in Isaac Sim to emulate elasticity, the real-world object's internal structure (e.g., whether it is hollow) cannot be inferred solely from its appearance, leading to inconsistencies between simulation and reality. Furthermore, the mesh generation module frequently failed in this task. The 3D generation model occasionally produced a rectangular cuboid instead of a cube, resulting in poor geometric alignment. Additionally, the result checking module sometimes misclassified scenarios where two cubes were merely placed adjacent to each other as successful stacking cases, which also contributed to the lower reported success rate.

For the third task, "put the blue cup standing with its opening upside on the right wooden block", mesh alignment turned out to be the dominant source of failure. The DINOv2 model often struggled to produce correct view-matching results during the rendering comparison stage, sometimes yielding completely reversed or perpendicular alignments with respect to the real-world observation. This issue is likely caused by the limited texture cues on the cup surface — being mostly uniform and glossy — which make the model highly sensitive to illumination differences between the real-world image and the rendering setup. Moreover, the task itself imposes strict orientation requirements, as the cup must stand perfectly upside down with minimal angular deviation. Combined with the limited reachable orientations of the robot arm at that specific position, the simulation-based strategy sampling sometimes failed to generate any feasible candidates that satisfy the grasp and placement constraints. As a result, even though the perception pipeline occasionally provided plausible digital twin model, the downstream sampling and execution stages were frequently unsuccessful.

To evaluate the effectiveness of the LLM-based sample result checker, we also computed its precision, recall, and F1-score on the three representative manipulation tasks, as summarized in Table 4.5. The checker determines whether the rendered outcome of a simulated strategy satisfies the given task prompt, effectively functioning as a semantic verifier between simulation and language-based intent.

The results indicate that the checker performs extremely well on the "put the banana into the basket" task, achieving an F1-score of 0.984. This is mainly because the visual cues of this task are clear and the task goal is unambiguous, allowing the LLM to easily identify success or failure from the rendered image. In contrast, the stacking cubes and cup-on-box tasks are more challenging. The stacking task often involves visually ambiguous cases, such as two cubes being close but not stably stacked or placed closely side by side rather than one on top of the other, leading to false positives and reduced precision. For the cup-on-box task, the main challenge lies in distinguishing the cup's orientation. Although the

prompt explicitly specifies “with its opening upside”, the LLM sometimes fails to differentiate between the cup’s opening and bottom, particularly under varying lighting or shading conditions in the rendered images. This leads to occasional false positives where a correctly placed but non-inverted cup is misinterpreted as successful.

Overall, the LLM-based checker does demonstrate strong generalization and semantic understanding across different object configurations. However, its sensitivity to visual ambiguity suggests that incorporating spatial reasoning or multimodal feedback (e.g., depth or contact information) could further improve its robustness.

Task	Precision	Recall	F1-score
Put the banana into the basket	0.993	0.977	0.984
Stack the green cube onto the yellow cube	0.644	0.829	0.699
Put the blue cup standing upside on the wooden box	0.590	0.615	0.600

Table 4.5: The precision, recall, and F1-score of the simulation result checking via LLM for three tasks.

5

Conclusion and Future Works

5.1. Conclusion

This thesis presented a novel framework that constructs an explicit world model to enable action sampling, simulation, and evaluation for open-world object manipulation tasks, conditioned on visual observations I and natural language task instructions C. Unlike end-to-end Vision-Language-Action (VLA) or imitation learning methods that require large amounts of task-specific action data, our approach leverages a physically grounded digital twin and simulator to reason about object dynamics and predict successful actions without any demonstrations.

The proposed system integrates open-set segmentation and grasping, 3D digital twin reconstruction, and simulation-based strategy sampling and evaluation within a unified pipeline. Experimental results demonstrate that the proposed two-stage mesh alignment pipeline is able to support the accuracy and robustness of digital twin reconstruction, providing more reliable object pose estimation. With the constructed explicit world model, the system is capable of performing dynamic reasoning and strategy evaluation across diverse manipulation tasks, and achieves successful transfer from simulation to real-world execution, proven by real robot experiments.

Furthermore, the detailed failure analysis highlights the contribution of each subsystem, including the effectiveness of the mesh alignment stage and the reliability of the LLM-based result checker in evaluating task outcomes. Overall, this research contributes toward bridging the gap between high-level semantic reasoning and physically grounded manipulation in open-world scenarios.

5.2. Limitation and Future Works

While the proposed framework demonstrates promising results, several limitations remain to be addressed in future research.

- **Simulation-to-real gap:** The most significant limitation arises from the discrepancy between simulated and real-world physics. Despite incorporating material estimation to improve realism, Isaac Sim's physical model still cannot fully capture complex object dynamics, such as compliance, surface friction, or hollow structures. Future work could integrate data-driven physical property estimation or real-to-sim calibration to reduce this gap.
- **Limited object types and dynamics:** This thesis focuses on rigid objects. Extending the framework to handle deformable, articulated, or fluid-like objects would significantly broaden its applicability in real-world tasks.
- **Integration of grasping and placing:** In this thesis, object grasping and placing are treated as two separate modules, which may lead to inconsistencies, as the optimal placing strategy often depends on the specific grasping outcome. Future work could address this limitation by integrating both stages into a unified framework that jointly considers grasp–place dependencies.
- **Temporal reasoning and task decomposition:** The current system evaluates single-step ma-

nipulation strategies. Incorporating temporal planning or hierarchical reasoning would enable the execution of multi-stage tasks involving sequential or conditional actions.

- **Efficiency and scalability:** The simulation-based strategy sampling process is not very efficient when scaling to more samples and complex scenes. Future work could explore learning-based surrogate models or differentiable simulators to accelerate evaluation.

In summary, this thesis contributes an important step toward building physically grounded world models for open-world robotic manipulation. Future work will focus on improving physical fidelity, extending the framework to complex object types and multi-step reasoning, and developing more efficient, generalizable systems that can learn and act robustly in the real world.

References

[1] Josh Achiam et al. “Gpt-4 technical report”. In: *arXiv preprint arXiv:2303.08774* (2023).

[2] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning (ICML)*. PMLR. 2021.

[3] Aakanksha Chowdhery et al. “Palm: Scaling language modeling with pathways”. In: *Journal of Machine Learning Research* (2023).

[4] Junnan Li et al. “Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation”. In: *International conference on machine learning (ICML)*. PMLR. 2022.

[5] Moo Jin Kim et al. “Openvla: An open-source vision-language-action model”. In: *arXiv preprint arXiv:2406.09246* (2024).

[6] Brianna Zitkovich et al. “Rt-2: Vision-language-action models transfer web knowledge to robotic control”. In: *Conference on Robot Learning (CoRL)*. PMLR. 2023.

[7] Kevin Black et al. “π: A Vision-Language-Action Flow Model for General Robot Control”. In: *CoRR* (2024).

[8] Physical Intelligence et al. “$\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization”. In: *arXiv preprint arXiv:2504.16054* (2025).

[9] Yi Li et al. “HAMSTER: Hierarchical Action Models for Open-World Robot Manipulation”. In: *The Thirteenth International Conference on Learning Representations (ICLR)*. 2025.

[10] Danijar Hafner et al. “Mastering Atari with Discrete World Models”. In: *International Conference on Learning Representations (ICLR)*. 2021.

[11] Danijar Hafner et al. “Dream to Control: Learning Behaviors by Latent Imagination”. In: *International Conference on Learning Representations (ICLR)*. 2020.

[12] Philipp Wu et al. “Daydreamer: World models for physical robot learning”. In: *Conference on robot learning (CoRL)*. PMLR. 2023.

[13] Mengjiao Yang et al. “Learning Interactive Real-World Simulators”. In: *CoRR* (2023).

[14] Jake Bruce et al. “Genie: Generative Interactive Environments”. In: *Forty-first International Conference on Machine Learning (ICML)*. 2024.

[15] Andreas Blattmann et al. “Stable Video Diffusion: Scaling Latent Video Diffusion Models to Large Datasets”. In: *CoRR* (2023).

[16] Weijie Kong et al. “HunyuanVideo: A Systematic Framework For Large Video Generative Models”. In: *CoRR* (2024).

[17] Xiaoxiao Long et al. “A Survey: Learning Embodied Intelligence from Physical Simulators and World Models”. In: *arXiv preprint arXiv:2507.00917* (2025).

[18] Taoran Jiang et al. “DexSim2Real^2: Building Explicit World Model for Precise Articulated Object Dexterous Manipulation”. In: *IEEE Transactions on Robotics* (2025).

[19] Matei Ciocarlie et al. “Towards reliable grasping and manipulation in household environments”. In: *Experimental Robotics: The 12th International Symposium on Experimental Robotics (ISER)*. Springer. 2014.

[20] Ian Lenz, Honglak Lee, and Ashutosh Saxena. “Deep learning for detecting robotic grasps”. In: *The International Journal of Robotics Research* (2015).

[21] Umar Asif, Jianbin Tang, and Stefan Harrer. “EnsembleNet: Improving Grasp Detection using an Ensemble of Convolutional Neural Networks.” In: *BMVC*. 2018.

[22] Umar Asif, Jianbin Tang, and Stefan Harrer. “Graspnet: An efficient convolutional neural network for real-time grasp detection for low-powered devices.” In: *IJCAI*. 2018.

[23] Yun Jiang, Stephen Moseson, and Ashutosh Saxena. “Efficient grasping from RGBD images: Learning using a new rectangle representation”. In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*. 2011.

[24] Fu-Jen Chu, Ruinian Xu, and Patricio A Vela. “Real-world multiobject, multigrasp detection”. In: *IEEE Robotics and Automation Letters* (2018).

[25] Hongzhuo Liang et al. “PointNetGPD: Detecting Grasp Configurations from Point Sets”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, May 2019.

[26] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. “6-dof graspnet: Variational grasp generation for object manipulation”. In: *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*. 2019.

[27] Andreas Ten Pas et al. “Grasp pose detection in point clouds”. In: *The International Journal of Robotics Research* (2017).

[28] Jacob Varley et al. “Generating multi-fingered robotic grasps via deep learning”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015.

[29] Michel Breyer et al. “Volumetric grasping network: Real-time 6 dof grasp detection in clutter”. In: *Conference on robot learning (CoRL)*. PMLR. 2021.

[30] Peiyuan Ni et al. “Pointnet++ grasping: Learning an end-to-end spatial grasp generation algorithm from sparse point clouds”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020.

[31] Yuzhe Qin et al. “S4g: Amodal single-view single-shot se (3) grasp detection in cluttered scenes”. In: *Conference on robot learning (CoRL)*. PMLR. 2020.

[32] Chenxi Wang et al. “Graspness discovery in clutters for fast and accurate grasp detection”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021.

[33] Liunian Harold Li et al. “Grounded language-image pre-training”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*. 2022.

[34] Shilong Liu et al. “Grounding dino: Marrying dino with grounded pre-training for open-set object detection”. In: *European conference on computer vision (ECCV)*. Springer. 2024.

[35] Hao Zhang et al. “DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection”. In: *The Eleventh International Conference on Learning Representations (ICLR)*. 2023.

[36] Tianhe Ren et al. *Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks*. 2024. arXiv: 2401.14159 [cs.CV].

[37] Alexander Kirillov et al. “Segment anything”. In: *Proceedings of the IEEE/CVF international conference on computer vision (ICCV)*. 2023.

[38] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies (NAACL-HLT), volume 1 (long and short papers)*. 2019.

[39] Aixin Liu et al. “DeepSeek-V3 Technical Report”. In: *CoRR* (2024).

[40] Hugo Touvron et al. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971 [cs.CL].

[41] OpenAI et al. *GPT-4o System Card*. 2024. arXiv: 2410.21276 [cs.CL].

[42] Abby O’Neill et al. “Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0”. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2024.

[43] Alexander Khazatsky et al. “DROID: A large-scale in-the-wild robot manipulation dataset”. In: *Robotics: Science and Systems (RSS)*. 2024.

[44] Homer Rich Walke et al. “Bridgedata v2: A dataset for robot learning at scale”. In: *Conference on Robot Learning*. PMLR. 2023.

[45] *NVIDIA. (n.d.) World models*. https://www.nvidia.com/en-us/glossary/world-models/. Accessed: 2025-10-20.

[46] Seyed S Mohammadi et al. “3DSGrasp: 3D Shape-Completion for Robotic Grasp”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023.

[47] Diego Hidalgo-Carvajal et al. “Anthropomorphic Grasping with Neural Object Shape Completion”. In: *IEEE Robotics and Automation Letters* 12 (2023).

[48] Federico Magistri et al. “Improving Robotic Fruit Harvesting Within Cluttered Environments Through 3D Shape Completion”. In: *IEEE Robotics and Automation Letters* (2024).

[49] Jeong Joon Park et al. “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.

[50] James J Kuffner and Steven M LaValle. “RRT-connect: An efficient approach to single-query path planning”. In: *Proceedings 2000 ICRA. Millennium conference. IEEE international conference on robotics and automation (ICRA). Symposia proceedings (Cat. No. 00CH37065)*. IEEE. 2000.

[51] L.E. Kavraki et al. “Probabilistic roadmaps for path planning in high-dimensional configuration spaces”. In: *IEEE Transactions on Robotics and Automation* (1996).

[52] Dmitry Berenson et al. “Manipulation planning on constraint manifolds”. In: *2009 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2009.

[53] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. “Task space regions: A framework for pose-constrained manipulation planning”. In: *The International Journal of Robotics Research* (2011).

[54] Zachary Kingston, Mark Moll, and Lydia E Kavraki. “Exploring implicit spaces for constrained sampling-based planning”. In: *The International Journal of Robotics Research* (2019).

[55] Jesse Haviland and Peter Corke. “NEO: A novel expeditious optimisation algorithm for reactive motion control of manipulators”. In: *IEEE Robotics and Automation Letters* (2021).

[56] Nathan D Ratliff et al. “Riemannian motion policies”. In: *arXiv preprint arXiv:1801.02854* (2018).

[57] Max Spahn, Martijn Wisse, and Javier Alonso-Mora. “Dynamic optimization fabrics for motion generation”. In: *IEEE Transactions on Robotics* (2023).

[58] Mohak Bhardwaj et al. “Storm: An integrated framework for fast joint-space model-predictive control for reactive manipulation”. In: *Conference on Robot Learning (CoRL)*. PMLR. 2022.

[59] Corrado Pezzato et al. “Sampling-based model predictive control leveraging parallelizable physics simulations”. In: *IEEE Robotics and Automation Letters* (2025).

[60] Mayank Mittal et al. “Articulated Object Interaction in Unknown Scenes with Whole-Body Mobile Manipulation”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2022.

[61] Johannes Pankert and Marco Hutter. “Perceptive model predictive control for continuous mobile manipulation”. In: *IEEE Robotics and Automation Letters* (2020).

[62] Adam Heins and Angela P. Schoellig. “Keep It Upright: Model Predictive Control for Nonprehensile Object Transportation With Obstacle Avoidance on a Mobile Manipulator”. In: *IEEE Robotics and Automation Letters* (Dec. 2023).

[63] Haoshu Fang et al. “AnyGrasp: Robust and Efficient Grasp Perception in Spatial and Temporal Domains”. In: *IEEE Trans. Robotics* (Oct. 2023).

[64] Ben Mildenhall et al. “Nerf: Representing scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM* (2021).

[65] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Transactions on Graphics* (2023).

[66] Binbin Xu, Andrew J Davison, and Stefan Leutenegger. “Learning to complete object shapes for object-level mapping in dynamic scenes”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022.

[67] Yue Pan et al. “Panoptic mapping with fruit completion and pose estimation for horticultural robots”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023.

[68] Zibo Zhao et al. “Hunyuan3D 2.0: Scaling Diffusion Models for High Resolution Textured 3D Assets Generation”. In: *CoRR* (2025).

[69] Maxime Oquab et al. “DINOv2: Learning Robust Visual Features without Supervision”. In: *Transactions on Machine Learning Research* (2024).

[70] Xinli Xu et al. “GaussianProperty: Integrating Physical Properties to 3D Gaussians with LMMs”. In: *arXiv preprint arXiv:2412.11258* (2024).