

# Scalable 3D Scene Understanding and Embodied Reasoning for Robotics

by

Shengjie Lin (林圣杰)

A thesis submitted  
in partial fulfillment of the requirements for  
the degree of

Doctor of Philosophy in Computer Science

at the

Toyota Technological Institute at Chicago  
Chicago, Illinois

September 2025

Thesis Committee:  
Matthew Walter (Thesis Advisor)  
Greg Shakhnarovich  
Vitor Guizilini

# Abstract

This thesis addresses the critical challenge of enabling robotic agents to perceive, understand, and intelligently interact with complex, large-scale 3D environments. We develop a comprehensive framework that progresses from foundational scene reconstruction to high-level embodied reasoning, enabling robots to perceive their physical surroundings and act on natural language instructions. The contributions are presented across three main chapters, each building upon the last to create a cohesive system for scalable 3D scene understanding and manipulation.

The first part of this work focuses on overcoming the limitations of representing spatially-extended environments. We introduce **NeRFuser**, a novel framework that constructs high-fidelity, large-scale 3D scene representations by registering and blending multiple, independently created Neural Radiance Fields (NeRFs). By operating without access to original training images or poses, NeRFuser provides a practical and scalable solution for creating detailed models of large areas, laying the essential groundwork for a robot’s spatial awareness.

Building on this static scene representation, the second part delves into understanding the dynamic properties of objects, a crucial capability for meaningful interaction. We present **SplArt**, a method for reconstructing articulated objects by observing them in just two distinct states. SPLART solves for the object’s kinematic properties, such as its articulation model, from sparse temporal data. This enables not only the accurate 3D reconstruction of dynamic objects but also provides the necessary understanding of their movable parts for subsequent robotic manipulation tasks.

The final part of this thesis bridges the gap between scene understanding and embodied intelligence by leveraging the reasoning capabilities of Large Language Models (LLMs), explored through two complementary systems. First, **Transcrib3D** empowers robots to ground complex, free-form referring



expressions by reasoning over a textual transcript of the 3D scene, enabling robust interpretation of human commands. Second, **Statler** introduces a state-maintaining LLM architecture that explicitly tracks the state of the world. This allows the robot to perform long-horizon, multi-step tasks by reasoning over both the user’s intent and a consistently updated world state. Together, these works demonstrate how LLMs can serve as a powerful reasoning engine for robotic agents, connecting high-level goals to low-level actions within a meticulously reconstructed and understood 3D world.

In essence, this thesis presents a unified pipeline that advances robotic perception and reasoning from the ground up. By first establishing a robust method for scalable 3D scene reconstruction and dynamic object understanding, we create a rich, detailed world model that integrates three-dimensional structure, articulated-object dynamics, and the world state that evolves through interaction. We then demonstrate that this model can be effectively utilized by LLM-based agents to understand human language and maintain state for complex, long-horizon tasks. The collective contributions pave the way for more capable and collaborative robotic systems that can seamlessly operate in human-centric environments.

# Acknowledgements

I owe my deepest gratitude to my advisor, Matthew Walter. Our weekly meetings anchored my doctoral studies: his high standards, clear thinking, and steady encouragement shaped how I do research. He offered not only technical guidance but also personal support when it mattered most. This dissertation reflects his mentorship at every stage.

I am grateful to my committee members, Greg Shakhnarovich and Vitor Guizilini, for their steady support and incisive advice during project discussions. Their questions and feedback sharpened the ideas in this thesis and raised the bar for the work.

I also thank my lab mates for the community we built—both the enlightening academic discussions that pushed the research forward and the real-life activities that kept things balanced. The hallway debates, paper clubs, code reviews, tea-time chats, and occasional off-campus escapes made this journey far better than it would have been alone.

My thanks go to the TTIC administrative staff who keep everything running smoothly. I especially thank Adam Bohlander for promptly and patiently resolving IT issues, and Mary Marre for her kindness and for seamlessly handling logistics—food, drinks, and all the details that make events work. Your behind-the-scenes efforts created the environment in which this research could happen.

I am indebted to my parents, Yiping Lin (林宜平) and Xiaoqing Yuan (袁小清), for unwavering love and encouragement. Their example of perseverance and optimism carried me through the inevitable setbacks.

Finally, to my wife, Haoyue Tang (汤皓玥): thank you for your patience, humor, and steady companionship through the final years of my PhD. Your support turned difficult stretches into finish lines. This accomplishment is as much yours as it is mine.

# Contents

|                                                              |             |
|--------------------------------------------------------------|-------------|
| <b>Abstract</b>                                              | <b>ii</b>   |
| <b>Acknowledgements</b>                                      | <b>iv</b>   |
| <b>List of Figures</b>                                       | <b>viii</b> |
| <b>List of Tables</b>                                        | <b>x</b>    |
| <b>List of Listings</b>                                      | <b>xi</b>   |
| <b>1 Introduction</b>                                        | <b>1</b>    |
| 1.1 Core Contributions . . . . .                             | 2           |
| 1.2 Thesis Outline . . . . .                                 | 3           |
| <b>2 Scalable Scene Reconstruction</b>                       | <b>5</b>    |
| 2.1 Introduction . . . . .                                   | 5           |
| 2.2 Related Work . . . . .                                   | 7           |
| 2.2.1 Neural Radiance Fields and Scalability . . . . .       | 7           |
| 2.2.2 NeRF Registration . . . . .                            | 8           |
| 2.2.3 NeRF Blending . . . . .                                | 9           |
| 2.3 Methodology . . . . .                                    | 10          |
| 2.3.1 Registration from NeRF Re-Rendering . . . . .          | 10          |
| 2.3.2 Sample-Based NeRF Blending . . . . .                   | 15          |
| 2.4 Experiments . . . . .                                    | 18          |
| 2.4.1 Datasets . . . . .                                     | 19          |
| 2.4.2 Evaluation of NeRF Registration and Blending . . . . . | 20          |
| 2.4.3 Quantitative Analysis of Registration . . . . .        | 20          |
| 2.4.4 Quantitative Analysis of Blending . . . . .            | 21          |

|          |                                                                    |           |
|----------|--------------------------------------------------------------------|-----------|
| 2.4.5    | Ablation Studies . . . . .                                         | 23        |
| 2.5      | Implications for 3D Gaussian Splatting . . . . .                   | 26        |
| 2.6      | Relation to Subsequent Work . . . . .                              | 28        |
| 2.7      | Conclusion . . . . .                                               | 29        |
| <b>3</b> | <b>Understanding Articulated Objects</b>                           | <b>31</b> |
| 3.1      | Introduction . . . . .                                             | 31        |
| 3.2      | Related Work . . . . .                                             | 34        |
| 3.2.1    | Data-Driven Articulation Learning . . . . .                        | 34        |
| 3.2.2    | Representations for 3D Object Reconstruction . . . . .             | 35        |
| 3.2.3    | Articulated Object Reconstruction . . . . .                        | 36        |
| 3.3      | Preliminaries . . . . .                                            | 37        |
| 3.4      | Methodology . . . . .                                              | 37        |
| 3.4.1    | Problem Formulation . . . . .                                      | 37        |
| 3.4.2    | Articulation Operator for Gaussian Primitives . . . . .            | 38        |
| 3.4.3    | Decoupled Multi-Stage Optimization . . . . .                       | 41        |
| 3.4.4    | Application to Real-World Data . . . . .                           | 46        |
| 3.5      | Experiments . . . . .                                              | 46        |
| 3.5.1    | Experimental Setup . . . . .                                       | 47        |
| 3.5.2    | Quantitative and Qualitative Results . . . . .                     | 50        |
| 3.5.3    | Ablation Studies . . . . .                                         | 52        |
| 3.5.4    | Robustness of Geometric Consistency . . . . .                      | 53        |
| 3.5.5    | Real-World Application . . . . .                                   | 54        |
| 3.6      | Discussion . . . . .                                               | 55        |
| 3.6.1    | Scope and Limitations . . . . .                                    | 55        |
| 3.6.2    | Future Directions . . . . .                                        | 55        |
| 3.6.3    | Relation to Subsequent Work . . . . .                              | 57        |
| 3.7      | Conclusion . . . . .                                               | 58        |
| <b>4</b> | <b>LLM-Powered Embodied Reasoning</b>                              | <b>60</b> |
| 4.1      | Introduction . . . . .                                             | 60        |
| 4.2      | Related Work . . . . .                                             | 63        |
| 4.2.1    | Large Language Models for Robotic Reasoning and Planning . . . . . | 63        |
| 4.2.2    | Grounding Language in 3D Scenes . . . . .                          | 64        |
| 4.2.3    | State Maintenance for Long-Horizon Reasoning . . . . .             | 65        |
| 4.3      | Methodology . . . . .                                              | 66        |

|          |                                                                                |            |
|----------|--------------------------------------------------------------------------------|------------|
| 4.3.1    | Transcrib3D: Grounding Referring Expressions via Scene Transcription . . . . . | 66         |
| 4.3.2    | Statler: State-Maintaining Models for Embodied Reasoning . . . . .             | 71         |
| 4.4      | Experiments . . . . .                                                          | 76         |
| 4.4.1    | Evaluating Transcrib3D: 3D Referring Expression Resolution . . . . .           | 76         |
| 4.4.2    | Evaluating Statler: Long-Horizon Embodied Reasoning . . . . .                  | 80         |
| 4.4.3    | Application to Real-World Robotic Manipulation . . . . .                       | 85         |
| 4.5      | Discussion and Limitations . . . . .                                           | 87         |
| 4.5.1    | Limitations of Transcrib3D . . . . .                                           | 87         |
| 4.5.2    | Limitations of Statler . . . . .                                               | 89         |
| 4.5.3    | Pathways to Integration . . . . .                                              | 89         |
| 4.5.4    | Relation to Subsequent Work . . . . .                                          | 91         |
| 4.6      | Conclusion . . . . .                                                           | 92         |
| <b>5</b> | <b>Conclusion</b> . . . . .                                                    | <b>94</b>  |
| 5.1      | Summary of Contributions . . . . .                                             | 94         |
| 5.2      | Broader Implications . . . . .                                                 | 95         |
| 5.3      | Future Directions . . . . .                                                    | 96         |
| 5.4      | Concluding Remarks . . . . .                                                   | 98         |
|          | <b>Bibliography</b> . . . . .                                                  | <b>99</b>  |
| <b>A</b> | <b>Supplementary Material for SPLART</b> . . . . .                             | <b>119</b> |
| A.1      | Additional Qualitative Results . . . . .                                       | 119        |
| A.2      | Per-Scene Quantitative Results . . . . .                                       | 119        |
| <b>B</b> | <b>Supplementary Material for Statler</b> . . . . .                            | <b>128</b> |
| B.1      | Example Evaluation Episodes . . . . .                                          | 128        |
| B.2      | Prompts . . . . .                                                              | 128        |
| B.3      | Example Failure Cases . . . . .                                                | 134        |
| B.3.1    | Pick-and-Place . . . . .                                                       | 134        |
| B.3.2    | Disinfection . . . . .                                                         | 134        |
| B.3.3    | Relative Weight Reasoning . . . . .                                            | 136        |

# List of Figures

|      |                                                                                      |    |
|------|--------------------------------------------------------------------------------------|----|
| 2.1  | NeRFuser framework overview . . . . .                                                | 6  |
| 2.2  | Distributions of training and re-rendering poses . . . . .                           | 12 |
| 2.3  | Comparison of NeRF rendering quality metrics . . . . .                               | 13 |
| 2.4  | Conceptual illustration of IDW-based blending variants . . . . .                     | 16 |
| 2.5  | Illustration of the ray sample merging process . . . . .                             | 17 |
| 2.6  | Qualitative comparison of NeRF blending methods . . . . .                            | 18 |
| 2.7  | NeRF blending comparison on the Mission Bay dataset . . . . .                        | 22 |
| 2.8  | NeRF registration error and time consumption . . . . .                               | 23 |
| 2.9  | Effect of re-rendering poses on NeRF registration . . . . .                          | 24 |
| 2.10 | Effect of the NeRF blending rate . . . . .                                           | 25 |
| 2.11 | Relative performance of <i>IDW-Sample</i> to <i>Nearest</i> . . . . .                | 26 |
| 3.1  | SPLART framework overview . . . . .                                                  | 32 |
| 3.2  | Transforming the view direction for the articulated Gaussian . . . . .               | 39 |
| 3.3  | SPLART methodology overview . . . . .                                                | 40 |
| 3.4  | Handling geometrically ambiguous parts . . . . .                                     | 44 |
| 3.5  | Handling partial observations . . . . .                                              | 45 |
| 3.6  | Comparison on part-level reconstruction and articulation estimation . . . . .        | 52 |
| 3.7  | State interpolation from the estimated articulation. . . . .                         | 55 |
| 3.8  | Analyzing the geometric consistency robustness for articulation estimation . . . . . | 56 |
| 3.9  | Real-world articulated object reconstructions . . . . .                              | 57 |
| 3.10 | Failure cases due to inherent ambiguities in articulation . . . . .                  | 58 |
| 4.1  | Transcrib3D framework overview . . . . .                                             | 61 |
| 4.2  | Statler framework overview . . . . .                                                 | 62 |
| 4.3  | Iterative code generation and reasoning process . . . . .                            | 69 |

|      |                                                                                                |     |
|------|------------------------------------------------------------------------------------------------|-----|
| 4.4  | Application of Transcrib3D to the real robot manipulation . .                                  | 72  |
| 4.5  | Model accuracies on the <i>three-cups-and-a-ball</i> shell game . . .                          | 75  |
| 4.6  | Comparisons between Transcrib3D and 3D-VisTA . . . . .                                         | 78  |
| 4.7  | The simulated domains used to evaluate Statler . . . . .                                       | 81  |
| 4.8  | LLM queries with and without a maintained world state . . .                                    | 82  |
| 4.9  | Simulation scenarios comparing Code-as-Policies and Statler .                                  | 83  |
| 4.10 | Grounding performance comparisons of CaP+MDETR and<br>CaP+Transcrib3D . . . . .                | 86  |
| 4.11 | Real robot experiment comparing Code-as-Policies and Statler                                   | 87  |
| 4.12 | Typical failure cases in Transcrib3D . . . . .                                                 | 88  |
| A.1  | Additional visualizations of state interpolation . . . . .                                     | 120 |
| A.2  | Additional comparisons of part-level reconstruction and artic-<br>ulation estimation . . . . . | 122 |

# List of Tables

|     |                                                                                           |     |
|-----|-------------------------------------------------------------------------------------------|-----|
| 2.1 | Blending results on Object-Centric Indoor Scenes . . . . .                                | 20  |
| 2.2 | ScanNet registration results . . . . .                                                    | 21  |
| 2.3 | Blending results on Mission Bay dataset. . . . .                                          | 22  |
| 3.1 | PARIS-PMS articulation and mesh reconstruction metrics . .                                | 50  |
| 3.2 | SPLART-PMS articulation metrics . . . . .                                                 | 51  |
| 3.3 | PARIS-PMS novel view synthesis metrics . . . . .                                          | 53  |
| 3.4 | SPLART-PMS novel view and articulation synthesis metrics .                                | 54  |
| 4.1 | Grounding accuracy on Nr3D and Sr3D . . . . .                                             | 77  |
| 4.2 | Grounding accuracy on ScanRefer . . . . .                                                 | 79  |
| 4.3 | Performance of fine-tuning models on NR3D . . . . .                                       | 80  |
| 4.4 | Episode and individual step success rates . . . . .                                       | 82  |
| 4.5 | Success rates for non-temporal and temporal queries . . . . .                             | 84  |
| 4.6 | Ablation episode (individual step) success rates . . . . .                                | 85  |
| A.1 | PARIS-PMS articulation and mesh reconstruction metrics . .                                | 121 |
| A.2 | SPLART-PMS articulation metrics on revolute scenes . . . . .                              | 123 |
| A.3 | SPLART-PMS articulation metrics on prismatic scenes . . . .                               | 124 |
| A.4 | PARIS-PMS novel view synthesis metrics . . . . .                                          | 125 |
| A.5 | SPLART-PMS novel view and articulation synthesis metrics<br>on revolute scenes . . . . .  | 126 |
| A.6 | SPLART-PMS novel view and articulation synthesis metrics<br>on prismatic scenes . . . . . | 127 |



# List of Listings

|     |                                                                                       |     |
|-----|---------------------------------------------------------------------------------------|-----|
| 4.1 | Example of self-reasoned correction for fine-tuning . . . . .                         | 71  |
| 4.2 | The prompt and desired output of the vanilla LLM . . . . .                            | 73  |
| 4.3 | The prompt and desired output of LLM+CoT . . . . .                                    | 73  |
| 4.4 | The prompt and desired output of LLM+State . . . . .                                  | 73  |
| 4.5 | World-state reader . . . . .                                                          | 74  |
| 4.6 | World-state writer . . . . .                                                          | 74  |
| B.1 | Example evaluation episode (Pick-and-Place domain) . . . . .                          | 129 |
| B.2 | Example evaluation episode (Disinfection domain) . . . . .                            | 130 |
| B.3 | Example evaluation episode (Relative Weight domain) . . . . .                         | 131 |
| B.4 | Prompt for Code-as-Policies (Disinfection domain) . . . . .                           | 132 |
| B.5 | Prompt for Statler (Disinfection domain) . . . . .                                    | 133 |
| B.6 | Generated states and code on an evaluation episode (Pick-and-Place domain) . . . . .  | 135 |
| B.7 | Generated states and code on an evaluation episode (Disinfection domain) . . . . .    | 137 |
| B.8 | Generated states and code on an evaluation episode (Relative Weight domain) . . . . . | 138 |

# Chapter 1

## Introduction

For robotic agents to become truly integrated into human-centric environments, they must possess the ability to perceive, comprehend, and interact with the world in a manner that is both intelligent and scalable. This requires more than just recognizing objects; it demands a deep understanding of the three-dimensional structure of large-scale spaces, the dynamic properties of the objects within them, and the nuances of human language used to direct action. Current robotic systems often struggle with these interconnected challenges. They may be limited to small, predefined workspaces, lack the ability to understand how objects can move or change, or fail to interpret complex, colloquial instructions from human collaborators. This thesis confronts these limitations directly by proposing a unified framework that builds a bridge from foundational 3D scene understanding to high-level, language-driven embodied reasoning.

The central challenge this work addresses is enabling a robot to build a rich, actionable model of its environment and then leverage that model to perform complex, long-horizon tasks specified through natural language. This problem can be decomposed into two fundamental questions: (1) How can a robot efficiently build a detailed and accurate 3D representation of its environment, especially one that extends beyond a single viewpoint? and (2) How can this representation be used to reason about the world’s state, understand dynamic object properties, and ground human language for intelligent manipulation?

To answer these questions, this thesis presents a series of contributions that advance the state of the art in robotic perception and reasoning. Our approach follows a logical progression, starting with the fundamental problem

of scene representation and culminating in sophisticated, language-enabled robotic behavior. First, we tackle the challenge of spatial scalability in 3D reconstruction. We then move to understanding the dynamic, articulated nature of objects, a critical step for meaningful interaction. Finally, we integrate the power of Large Language Models (LLMs) to empower the robotic agent to reason about its world, understand complex commands, and maintain a consistent understanding of the environment over time.

## 1.1 Core Contributions

The journey toward capable robotic assistants begins with a robust model of the physical world, as a robot’s ability to act is fundamentally constrained by its ability to accurately perceive its surroundings. This thesis constructs a comprehensive pipeline that progressively builds a *world model*—by which we mean the agent’s internal representation of its physical environment, encompassing its three-dimensional structure, the kinematics of the articulated objects within it, and the world state that evolves through interaction, rather than a predictive forward model in the generative-modeling sense. We build this representation in layers—a scalable spatial map of the environment, an understanding of how its articulated objects move, and an explicit world state that the agent updates as it acts—and use the resulting model to ground language commands and support high-level reasoning for intelligent action.

The first pillar of our research addresses the need for 3D scene representations that are not only high-fidelity but also scalable to large, real-world environments. While Neural Radiance Fields (NeRFs) [30] excel at synthesizing photorealistic views, their expressivity is fundamentally constrained by the capacity of a single model, making them ill-suited for representing expansive areas. As detailed in Chapter 2, we introduce **NeRFuser**, a framework that overcomes this by composing large-scale scenes from multiple, independently created NeRFs [137]. Critically, NeRFuser operates without access to the original training images or camera poses, making it highly practical for decentralized mapping scenarios. It achieves this through a two-stage process: a novel *registration from re-rendering* technique to robustly align the NeRFs in a common coordinate frame, followed by a fine-grained, *sample-based blending* method that produces seamlessly integrated views. This work establishes the foundational capability for a robot to build a comprehensive and scalable spatial map of its environment.

A static scene representation, however, is insufficient for meaningful interaction. Robots must also understand that objects have functional properties defined by their ability to move. The second part of this thesis, presented in Chapter 3, focuses on the reconstruction and understanding of articulated objects. Reconstructing such objects is challenging, yet crucial for manipulation. We introduce **SplArt**, a self-supervised and category-agnostic framework that reconstructs an articulated object and infers its kinematic properties from sparse observations of just two distinct end states [171]. Leveraging 3D Gaussian Splatting (3DGS) [104] for real-time, high-fidelity rendering, SPLART augments each Gaussian with a differentiable *mobility parameter*. This innovation, combined with a multi-stage optimization strategy and geometric self-supervision, allows the system to robustly segment the object into its static and mobile parts and accurately estimate its articulation model without requiring 3D annotations or category-specific priors.

With the spatial and kinematic layers of our world model in place, the final step is to connect them to high-level embodied intelligence using Large Language Models (LLMs). Chapter 4 explores this through two complementary systems. First, **Transcrib3D** addresses the critical task of grounding natural language in the 3D world [138]. It sidesteps the data scarcity problem of traditional multi-modal methods by using *text* as a unifying medium. A 3D object detector first generates a textual “transcript” of the scene, which an LLM then reasons over to resolve complex referring expressions. Second, **Statler** introduces a state-maintaining LLM architecture to tackle long-horizon tasks [162]. It employs a *world-state reader* to generate actions based on the current state and a *world-state writer* to update the state after execution. By making the world state explicit, Statler enables a robot to reason over a consistently updated view of its environment. Together, these systems provide a powerful reasoning engine, translating high-level human goals into concrete actions within the reconstructed 3D world.

## 1.2 Thesis Outline

This thesis is structured into three main chapters, each presenting a core contribution to the overarching goal of scalable 3D scene understanding and embodied reasoning for robotics.

- Chapter 2 introduces NeRFuser, a novel framework for creating high-fidelity, large-scale 3D scene representations. We detail our proposed

methods for registering and blending multiple pre-trained NeRFs without requiring access to the original training images or poses. This work establishes the foundational capability for a robot to build a comprehensive spatial map of its environment.

- Chapter 3 presents SPLART, a method for jointly reconstructing articulated objects and estimating their kinematic properties from temporally sparse observations. We demonstrate how, by observing an object in just two states, our method can infer its articulation model, enabling both photorealistic rendering and a functional understanding necessary for robotic manipulation.
- Chapter 4 bridges scene understanding with robotic intelligence by leveraging Large Language Models. We present two systems: Transcrib3D, which grounds complex referring expressions by reasoning over a textual description of the 3D scene, and Statler, which uses an LLM to maintain and reason over the world state for long-horizon tasks.

Finally, Chapter 5 concludes the thesis by summarizing the key contributions and discussing their broader implications. We will also outline promising directions for future research, exploring how these foundational components can be extended to create even more capable, collaborative, and intelligent robotic systems that can seamlessly operate in and alongside human society. In essence, this thesis presents a unified pipeline that advances robotic perception and reasoning from the ground up, paving the way for the next generation of embodied AI.

# Chapter 2

## Spatially-Scalable Scene Reconstruction with NeRFuser

### 2.1 Introduction

A robot’s ability to perform meaningful tasks in the physical world is predicated on its capacity to build and maintain an accurate internal model of its environment. This world model is the foundation that all subsequent reasoning, planning, and interaction depend on. Without a detailed spatial understanding, a robot is effectively blind, severely limiting its ability to plan deliberate action. Therefore, the first and most critical step toward enabling sophisticated embodied intelligence is to solve the problem of scalable, high-fidelity 3D scene reconstruction.

Recent years have seen a paradigm shift in view synthesis and scene representation with the advent of Neural Radiance Fields (NeRFs) [30]. By encoding a scene’s geometry and appearance within the weights of a neural network, NeRFs can render photorealistic novel views from a sparse set of input images, capturing complex light and view-dependent effects with remarkable fidelity. This powerful, continuous, and memory-efficient representation holds immense promise for robotics, offering a significant leap beyond traditional discrete representations like point-clouds or meshes.

However, the very architecture that gives NeRF its expressive power also imposes a fundamental limitation: **scalability**. A single neural network has a finite model capacity, making it inherently ill-suited for representing large, complex, or spatially-extended environments. While a single NeRF

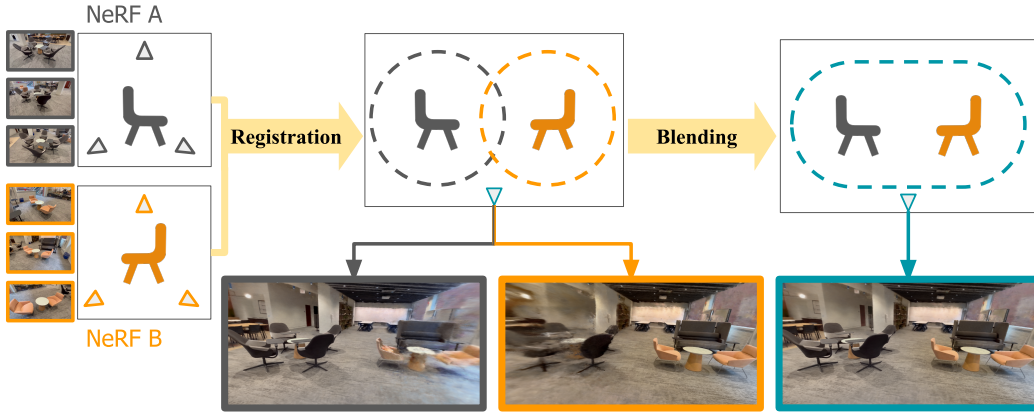


Figure 2.1: Given two independently constructed NeRFs, NeRF A and NeRF B, our method, NeRFuser, synthesizes images from *novel viewpoints*. It excels in views with sparse coverage from either input, producing results (*lower right*) that are qualitatively superior to the original NeRFs (A, B) and other baselines.

can faithfully reconstruct a tabletop object or a small room, its performance degrades significantly when tasked with modeling an entire building floor, a city block, or an expansive outdoor area. This scalability challenge presents a major roadblock to deploying NeRF-based representations on robots that must operate in and navigate through large, real-world spaces. While some approaches have sought to partition scenes into manageable chunks [77], they often rely on having a complete, globally consistent set of training data available from the outset.

This chapter introduces **NeRFuser**,<sup>1</sup> a novel framework designed to directly address this challenge by enabling the construction of large-scale scene representations from multiple, independently created NeRFs [137] (see Figure 2.1). The core premise of our approach is to treat individual, pre-trained NeRFs as modular building blocks, or “sub-maps,” which can be seamlessly stitched together to form a cohesive and expansive scene representation. Crucially, NeRFuser operates under a practical and challenging constraint that mirrors real-world multi-agent or decentralized mapping scenarios: we assume access only to the trained NeRF models themselves, *without any of the original training images or their associated camera poses*.

<sup>1</sup>Code is publicly available at <https://github.com/ripl/nerfuser>.

To achieve this, NeRFuser introduces a two-stage pipeline. First, we propose a novel *registration from re-rendering* technique that leverages the view-synthesis capability of the NeRFs themselves to generate new images that can be used to solve for the relative transformations (including scale) between them. This allows us to align disparate NeRFs into a common coordinate frame without needing their original training data. Second, once aligned, we employ a fine-grained, *sample-based blending* method that intelligently merges information from multiple NeRFs at the volumetric ray-sample level, producing renderings that are not only seamless but often superior in quality to those from any single source NeRF.

By composing scenes in this manner, NeRFuser provides the essential first component of our thesis: a practical and effective method for building the scalable, high-fidelity 3D scene representation required for a robot to possess true spatial awareness. This layer of scene understanding is the foundation on which the capabilities developed in subsequent chapters—understanding dynamic objects and performing high-level embodied reasoning—will be built.

## 2.2 Related Work

The core contribution of NeRFuser—scalable scene reconstruction by composing multiple neural radiance fields—is situated at the intersection of several active research areas. This section reviews the foundational work on Neural Radiance Fields (NeRFs), examines the specific challenges and prior approaches to scalability, and discusses the related problems of NeRF registration and blending.

### 2.2.1 Neural Radiance Fields and Scalability

A Neural Radiance Field (NeRF) [30] is an implicit neural representation that encodes a 3D scene’s geometry and appearance into the weights of a multilayer perceptron (MLP). By querying the network with 3D coordinates and a viewing direction, it outputs a volumetric density and an emitted radiance. These values can then be integrated along camera rays using volumetric rendering techniques to synthesize photorealistic novel views. The remarkable quality of NeRF has inspired a torrent of subsequent research aimed at improving its fidelity, training speed, and efficiency [33, 53, 59, 72, 76, 119].

Despite these advancements, a fundamental limitation of the original



NeRF formulation is its scalability. A single MLP has a finite capacity, which restricts its ability to represent large, complex, and spatially-extended environments. While some methods have demonstrated success in modeling large scenes with a single, high-capacity model, often aided by specialized data structures or positional encodings [72, 86], this approach can be computationally demanding and still faces inherent capacity limits.

To address this, a significant line of work has explored decomposing large scenes into smaller, more manageable parts, each represented by its own model. This strategy has been shown to improve both reconstruction quality and computational efficiency [77]. Methods like BungeeNeRF [81] and iMAP [45] employ progressive training schemes, while others utilize carefully designed data structures such as grids or trees to manage the complexity [84, 165]. Several works have demonstrated that a collection of many smaller models can outperform a single monolithic model while using a similar number of parameters [43, 77, 80]. However, these approaches typically assume that all training images and their corresponding camera poses are available in a globally consistent coordinate frame from the outset. Our work, NeRFuser, provides a novel and practical alternative for handling multiple small models, specifically targeting scenarios where such information is unavailable.

More recently, explicit representations like 3D Gaussian Splatting (3DGS) have offered new avenues for scalability. Methods such as VastGaussian [148] and Scaffold-GS [151] leverage the efficiency of Gaussian primitives to reconstruct massive scenes, while STORM [179] extends these capabilities to dynamic, spatio-temporal outdoor environments. While these approaches shift away from implicit MLPs, the core challenge of managing large-scale data remains relevant, and the principles of modular composition explored in NeRFuser offer valuable insights for these emerging representations.

### 2.2.2 NeRF Registration

To jointly utilize multiple NeRFs that have been trained independently, one must first solve the **NeRF Registration** problem: determining the relative rigid transformation (and scale) between their respective coordinate systems. This is a non-trivial task because NeRFs are typically trained using camera poses obtained from Structure-from-Motion (SfM) pipelines [5, 6]. These SfM methods, which rely on matching local features [12, 15, 20, 21, 31], are inherently scale-agnostic. Consequently, each NeRF is defined in its own arbitrary coordinate frame with an arbitrary scale.

NeRF registration differs from the related problem of “NeRF Inversion” [50, 106], which focuses on estimating the 6-DoF camera pose of a single image relative to a pre-trained NeRF. This camera-to-NeRF inversion capability has been successfully applied to robotic localization tasks [51, 111, 133]. Other related works jointly optimize the NeRF representation along with camera poses and intrinsics [37, 38, 48]. In contrast, NeRFuser operates on pre-trained, fixed NeRF models and uses SfM on newly rendered images, rather than modifying the NeRFs themselves.

The field of NeRF registration is nascent, with only a few prior works. Most of these approaches [75, 95, 103] rely exclusively on geometric cues, such as by extracting density fields or surfaces from the NeRFs and performing point-cloud-style registration. This fails to leverage the rich photometric information captured by the radiance field. DReg-NeRF [88] learns to predict 3D correspondences but suffers a performance drop on out-of-distribution scenes. Furthermore, both DReg-NeRF and nerf2nerf [95] are limited to registering only two NeRFs at a time, require known scale, and can fail on unbounded, large-scale scenes. Nerf2nerf also requires a reasonable initial alignment provided by a human. LENS [71] uses a NeRF to render additional images for training a scene-specific pose regression network, but this requires access to the original training data. NeRFuser distinguishes itself by being able to register multiple NeRFs simultaneously, recovering scale, and operating on large-scale scenes without human annotation or access to the original training data.

Recent advancements have also begun to address registration for explicit representations. For instance, GaussReg [132] introduces a fast 3D registration framework for Gaussian Splatting, utilizing a coarse-to-fine strategy that mirrors the hierarchical approaches used in point cloud registration. While NeRFuser targets implicit fields where explicit geometry is harder to access, the underlying need for robust, scale-aware alignment is shared across both.

### 2.2.3 NeRF Blending

Once multiple NeRFs are aligned in a common coordinate frame, the next critical challenge is to **blend** their predictions in order to render seamless and high-quality novel views. This involves intelligently merging information from overlapping NeRFs.

Prior works have explored different blending strategies. Nerflets [128] decomposes a scene into ellipsoidal regions, but assumes all NeRFs have access

to all training images. Blended-NeRF [96] focuses on generative applications rather than scene reconstruction. The most relevant prior work is Block-NeRF [77], which proposes blending NeRFs in either an image-wise or a pixel-wise fashion. To weight the contributions of different NeRFs, Block-NeRF considers two approaches: predicted visibility and inverse distance weighting (IDW). The visibility-based approach requires training an auxiliary network to predict which training cameras can see a given point, a process that requires access to the original training poses. This is not feasible in the NeRFuser setting. Block-NeRF ultimately adopts an IDW-based approach for its temporal consistency. Our work builds upon this idea but proposes a more fine-grained, sample-based blending technique that operates on the volumetric ray samples, which is more native to NeRFs’ rendering process.

Hybrid approaches have also emerged to combine the strengths of different representations. RadSplat [174] and NeuralGS [176] propose blending implicit neural fields with explicit Gaussian splats, aiming to achieve both the rendering speed of 3DGS and the compactness or regularization benefits of NeRFs. These works highlight the growing trend towards unifying distinct scene representations.

## 2.3 Methodology

The NeRFuser framework is designed to construct a cohesive, large-scale scene representation from a collection of independently trained NeRFs. This is achieved through a two-stage pipeline that first registers the NeRFs into a common coordinate system and then blends their volumetric information to synthesize novel views. This section details the core technical contributions of each stage: our *registration from re-rendering* technique and our fine-grained, *sample-based blending* algorithm. For the sake of clarity, we will describe the process for a pair of NeRFs, denoted NeRF A and NeRF B, though the methodology extends straightforwardly to multiple NeRFs.

### 2.3.1 Registration from NeRF Re-Rendering

The first fundamental challenge is to solve for the relative spatial transformation between the input NeRFs. Since each NeRF is trained independently, often using poses derived from a scale-agnostic SfM pipeline, they exist in their own local coordinate systems. Our goal is to find the full similarity

transformation,  ${}^A\mathbf{T}_B \in \text{Sim}(3)$ , that aligns a 3D point  $\mathbf{p}_B$  from NeRF B’s coordinate frame to its corresponding point  $\mathbf{p}_A$  in NeRF A’s frame, such that  $\mathbf{p}_A = {}^A\mathbf{T}_B\mathbf{p}_B$ . We achieve this without access to the original training images or poses by leveraging the NeRFs’ own view synthesis capabilities.

### Overview of the Registration Process

The core insight of our registration method is to use the NeRFs themselves as image generators. We strategically sample a set of new camera poses and use each NeRF to render, or “re-render,” a collection of images from these poses. With these newly synthesized images from all source NeRFs, we can employ an off-the-shelf, 2D feature-based Structure-from-Motion (SfM) pipeline [12, 31]. The SfM process recovers the poses of all the re-rendered images within a single, shared coordinate system. Since the local pose used to render each image is known, we can then compute the transformation from this shared SfM frame to each NeRF’s local frame, and consequently, the relative transformation between the NeRFs themselves.

### Pose Sampling Strategy for Re-Rendering

While we do not assume access to the original training poses, we rely on the standard practice of normalizing these poses during NeRF training, as established by popular frameworks like `nerfstudio` [119]. This convention typically involves: (1) centering the poses so the origin is at their mean translation; (2) rotating the coordinate system so the average “up” direction aligns with the positive  $z$ -axis; and (3) uniformly scaling the scene so all camera positions lie within a unit cube (i.e.,  $[-1, 1]^3$ ). Based on these conventions, we adopt a simple yet effective sampling strategy. We uniformly sample camera poses on the upper hemisphere of a unit sphere centered at the origin. This ensures that the re-rendered views are likely to capture meaningful content within the bounds of each NeRF’s well-reconstructed area. See Figure 2.2 for a visualization of the pose sampling strategy.

### Distant Accumulation for Pose Filtering

Not all sampled poses will produce high-quality renderings, especially those near the periphery of a NeRF’s training data. While SfM is robust to some degree of image noise, filtering out poor-quality renderings beforehand improves both the speed and accuracy of the registration process.

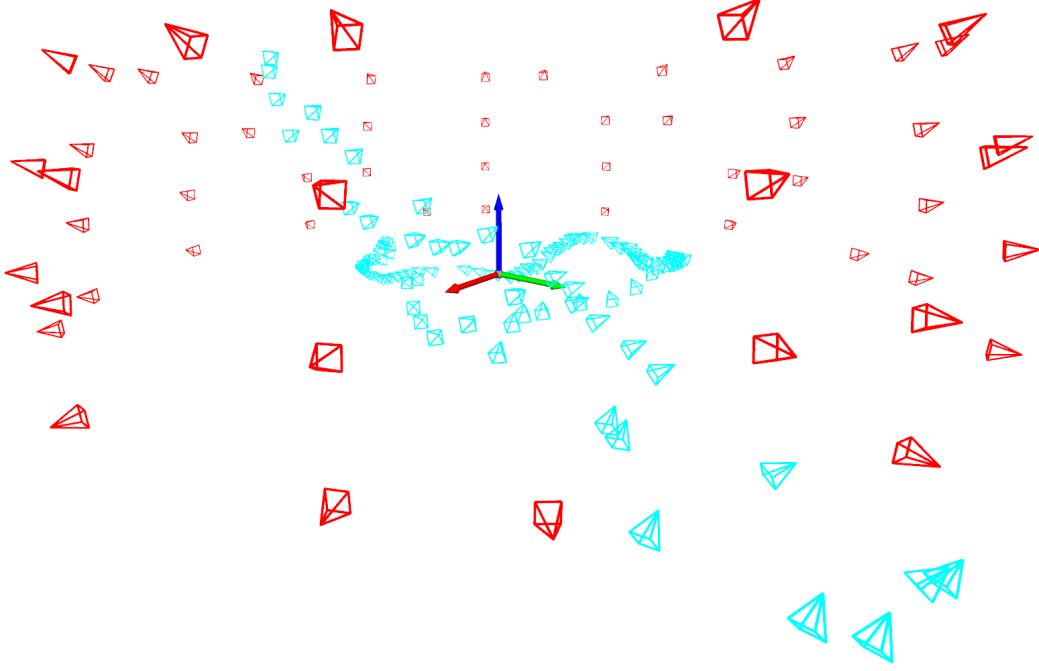


Figure 2.2: Distributions of **training poses** and **sampled poses** on a ScanNet scene used for NeRF registration. Sample poses are less cluttered than training ones that come from a handheld camera trajectory, which results in a wider baseline for easier SfM.

A common metric for NeRF ray quality is accumulated opacity, or simply *accumulation*, which measures the probability that a ray terminates. Low accumulation values often indicate under-trained regions of the scene. However, we still observe two failure modes: (1) when a ray is under-trained, its termination probability mass tends to be distributed broadly and evenly along its length, and (2) renderings from cameras positioned inside or extremely close to an object are often distorted and unhelpful for feature matching.

To address these issues, we introduce a novel ray quality measure called **Distant Accumulation**, defined as:

$$q_d = \int_d^\infty T(t)\sigma(t)dt, \quad (2.1)$$

where  $T(t)$  is the transmittance and  $\sigma(t)$  is the volume density at distance  $t$  along the ray, as defined in the original NeRF paper [30]. The parameter

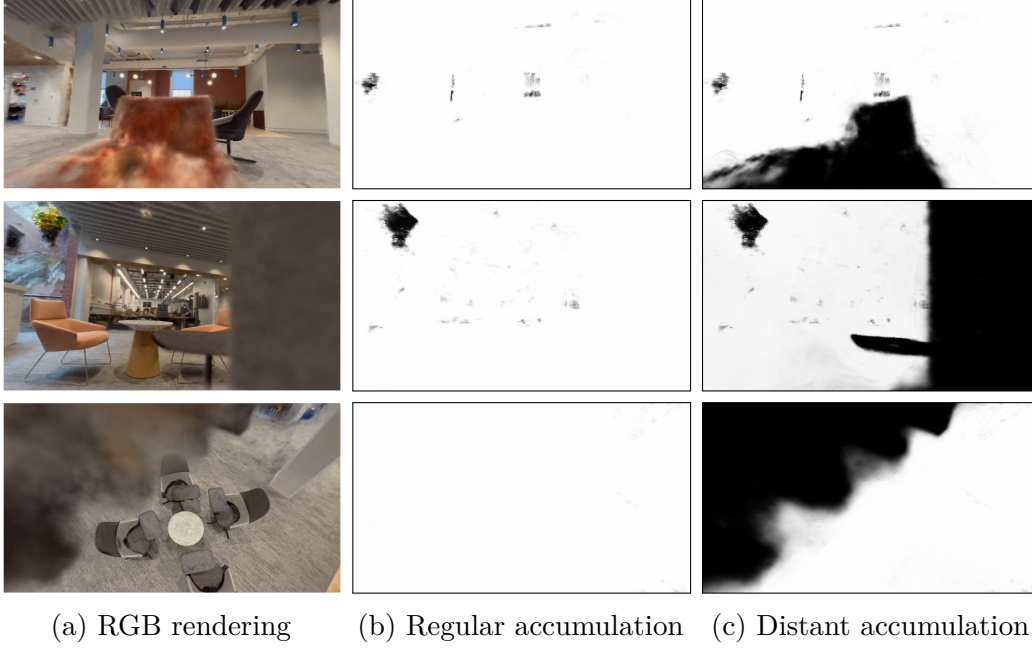


Figure 2.3: Comparison of NeRF rendering quality metrics. Column (a) shows RGB renderings. Column (b) shows that regular accumulation is often close to 1, failing to identify poor renderings. Column (c) demonstrates that our proposed distant accumulation ( $d = 0.3$ ) effectively highlights poorly reconstructed regions as darker areas, providing a reliable filter.

$d$  is a near-plane distance threshold. We use the mean distant accumulation over all pixels as a holistic image quality score.

Intuitively, this measure penalizes rays in two ways: (1) If a ray is under-trained and its termination probability is spread out, the integral from  $d$  to infinity will be substantially less than 1. (2) If the camera is too close to a surface, most rays will terminate at a distance less than  $d$ , resulting in a very low  $q_d$ . As illustrated in Figure 2.3, distant accumulation provides a much clearer signal for discriminating poor-quality renderings compared to regular accumulation. We use this metric to filter out unsuitable re-rendered images before feeding them into the SfM pipeline.

### Derivation of the Sim(3) Transformation

**Procedure and Notation.** Let  ${}^A\mathbf{G}_{A_i} \in \text{SE}(3)$  denote the sampling pose used to render image  $I_{A_i}$  from NeRF A—i.e., the pose matrix that transforms a point from camera  $A_i$ ’s frame to NeRF A’s frame. Sampled poses and images for NeRF B are defined analogously:  ${}^B\mathbf{G}_{B_i}$  and  $I_{B_i}$ . Feeding the combined image set  $\{I_{A_i}\}_i \cup \{I_{B_i}\}_i$  to SfM returns recovered poses  $\{{}^C\mathbf{G}_{A_i}\}_i$  and  $\{{}^C\mathbf{G}_{B_i}\}_i$  in a shared coordinate frame C, where  ${}^C\mathbf{G}_{A_i} \in \text{SE}(3)$  transforms a point from camera  $A_i$ ’s frame to frame C. Because SfM is scale-agnostic, these  $\text{SE}(3)$  poses are expressed in frame C’s arbitrary unit of length, motivating the scale-recovery step below.

**Recovering the Scale.** Let  ${}^C\mathbf{S}_A = \text{diag}({}^C s_A, {}^C s_A, {}^C s_A, 1)$  be the diagonal matrix representing the scale transformation from NeRF A’s frame to the common frame C. The scalar  ${}^C s_A$  is the scale factor we aim to recover. Consider the relative pose  ${}^j\mathbf{G}_i \in \text{SE}(3)$  between two cameras,  $A_i$  and  $A_j$ . This relative pose can be computed in the common frame C or derived from the original poses in NeRF A’s frame, related by the scale factor:

$$\begin{aligned} {}^j\mathbf{G}_i &= ({}^C\mathbf{G}_{A_j})^{-1} {}^C\mathbf{G}_{A_i} && \text{(using frame C as bridge)} \\ &= {}^C\mathbf{S}_A ({}^A\mathbf{G}_{A_j})^{-1} {}^A\mathbf{G}_{A_i} ({}^C\mathbf{S}_A)^{-1} && \text{(using frame A as bridge)} \end{aligned} \quad (2.2)$$

By decomposing the pose matrices into rotation and translation components, e.g.,  ${}^C\mathbf{G}_{A_i} = \begin{bmatrix} {}^C\mathbf{R}_{A_i} & {}^C\mathbf{t}_{A_i} \\ \mathbf{0} & 1 \end{bmatrix}$ , Equation (2.2) can be expanded to:

$$\begin{aligned} &\begin{bmatrix} ({}^C\mathbf{R}_{A_j})^\top {}^C\mathbf{R}_{A_i} & ({}^C\mathbf{R}_{A_j})^\top ({}^C\mathbf{t}_{A_i} - {}^C\mathbf{t}_{A_j}) \\ \mathbf{0} & 1 \end{bmatrix} \\ &= \begin{bmatrix} ({}^A\mathbf{R}_{A_j})^\top {}^A\mathbf{R}_{A_i} & {}^C s_A ({}^A\mathbf{R}_{A_j})^\top ({}^A\mathbf{t}_{A_i} - {}^A\mathbf{t}_{A_j}) \\ \mathbf{0} & 1 \end{bmatrix} \end{aligned} \quad (2.3)$$

The rotation components on both sides are equal, as expected. By equating the L2 norm of the translation vectors from both sides of Equation (2.3), we can solve for the unknown scale factor  ${}^C s_A$ :

$${}^C s_A = \frac{\|{}^C\mathbf{t}_{A_i} - {}^C\mathbf{t}_{A_j}\|_2}{\|{}^A\mathbf{t}_{A_i} - {}^A\mathbf{t}_{A_j}\|_2} \quad (2.4)$$

In practice, to obtain a robust estimate, we compute this ratio for all available pairs of indices  $(i, j)$  and take the median value. The scale factor  ${}^C s_B$  for NeRF B is recovered in an identical manner.

**Recovering the Full Transformation.** Once the scale factors are known, we can determine the full similarity transformation  ${}^C\mathbf{T}_A \in \text{Sim}(3)$  from NeRF A’s frame to the common frame C. Using any camera  $A_i$  as a bridge, the transformation is given by:

$${}^C\mathbf{T}_A = {}^C\mathbf{G}_{A_i} {}^C\mathbf{S}_A ({}^A\mathbf{G}_{A_i})^{-1} \quad (2.5)$$

To ensure robustness, we compute  ${}^C\mathbf{T}_A$  for all available camera indices  $i$  and select the valid  $\text{Sim}(3)$  transformation that is closest to the median value. After similarly recovering  ${}^C\mathbf{T}_B$ , the final  $\text{Sim}(3)$  transformation from NeRF B to NeRF A is computed as  ${}^A\mathbf{T}_B = ({}^C\mathbf{T}_A)^{-1} {}^C\mathbf{T}_B$ .

### 2.3.2 Sample-Based NeRF Blending

Once the NeRFs are aligned in a common coordinate frame, the final step is to blend their predictions to synthesize high-quality, seamless novel views. This process involves answering two key questions:

- (i) **What to blend?** At what level of granularity should the information from multiple NeRFs be combined?
- (ii) **How to blend?** How should the contribution, or weight, of each NeRF be determined at that level?

Prior work, such as Block-NeRF [77], explored blending at the image level (IDW-2D) or pixel level (IDW-3D), using inverse distance weighting (IDW) based on the distance from the camera or expected depth to the center of each NeRF. However, these approaches do not fully leverage the volumetric nature of NeRF rendering.

Recognizing that NeRF rendering is fundamentally a process of integrating information from samples along a ray, we propose a more fine-grained approach: **sample-based inverse distance weighting (IDW-Sample)**. Instead of computing blending weights per-image or per-pixel, we compute them for each individual volumetric sample along a ray. This allows for a smoother, more physically-grounded fusion of information, which is particularly effective in regions of overlap where one NeRF may be more confident or accurate than another. Figure 2.4 illustrates the conceptual difference between these blending strategies.



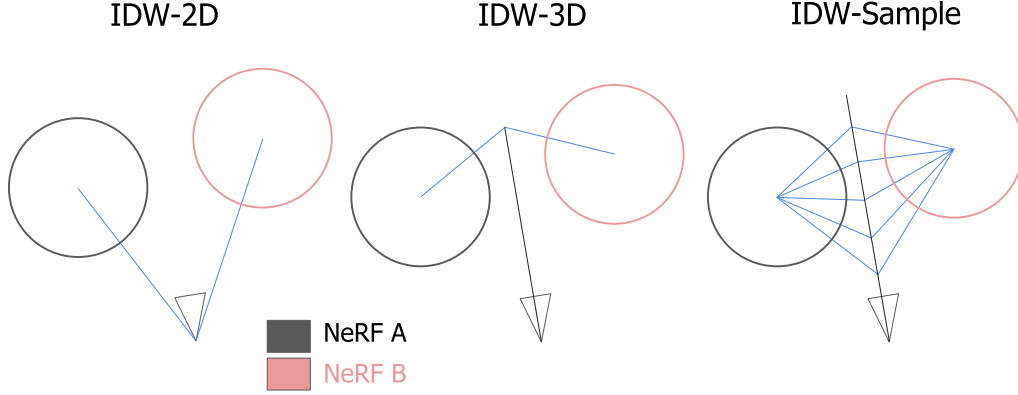


Figure 2.4: A conceptual illustration of different IDW-based blending methods. **IDW-2D** computes a single weight per NeRF based on camera distance. **IDW-3D** computes weights per-pixel based on the expected depth point. Our proposed **IDW-Sample** computes a unique weight for each volumetric sample along the ray, enabling a more fine-grained and accurate blend.

### Proximity Test

To maintain computational efficiency, we first perform a *proximity test* to determine which NeRFs are relevant for rendering a given pose. With all NeRFs registered in a common frame, we calculate the distance from the camera’s optical center to the origin of each NeRF. We then filter out NeRFs that are significantly farther away than the closest one, based on a distance ratio threshold  $\tau$ . This ensures that only the most relevant NeRFs contribute to the rendering of any given view.

### Merging Ray Samples

Each NeRF may use a different sampling strategy along a ray, often adaptively placing samples in regions of high density. To combine their predictions, we must first merge their distinct sets of ray samples into a single, unified set. Given the samples  $\{(t_k^A, \delta_k^A)\}_k$  from NeRF A and  $\{(t_k^B, \delta_k^B)\}_k$  from NeRF B, where  $t$  is the distance along the ray and  $\delta$  is the interval length, we create a new, sorted set of intervals  $\{(\bar{t}_k, \bar{\delta}_k)\}_k$  that covers all orig-

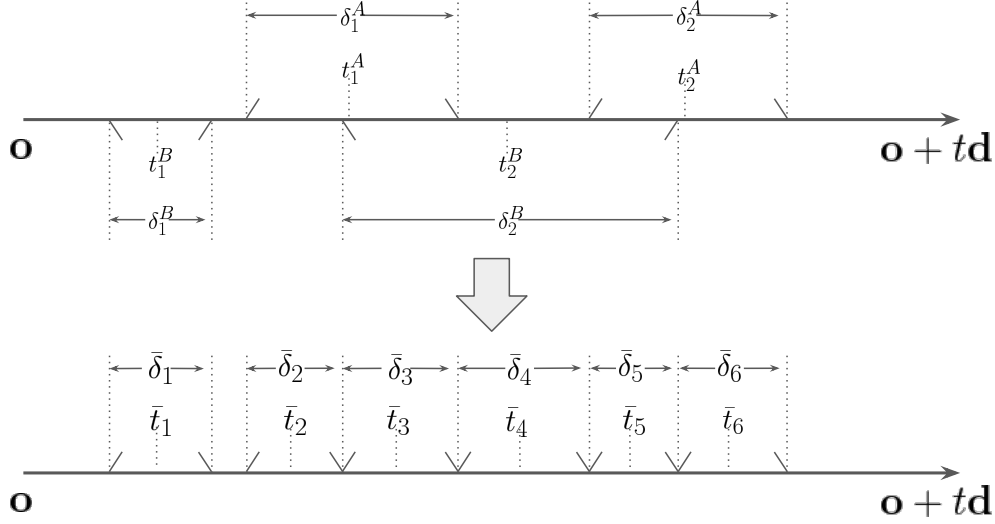


Figure 2.5: Illustration of the ray sample merging process. The distinct, irregular samples proposed by NeRF A (top) and NeRF B (middle) are combined into a single, unified set of sorted sample intervals (bottom).

inal samples, as shown in Figure 2.5. For each new interval in the merged set, we then compute the corresponding color and termination probability from each source NeRF.

### The Blending Process

With the merged set of ray samples, we compute the final pixel color by performing a weighted sum of the contributions from each NeRF at each sample point. For a given ray  $\mathbf{r}$  originating at  $\mathbf{o}$  with direction  $\mathbf{d}$ , and a merged sample  $k$  at distance  $\bar{t}_k$ , the blending weight  $w_{i,k}$  for NeRF $_i$  is computed using IDW:

$$w_{i,k} \propto d_{i,k}^{-\gamma}, \quad \text{where} \quad d_{i,k} = \|\mathbf{x}_i - (\mathbf{o} + \bar{t}_k \mathbf{d})\|_2. \quad (2.6)$$

Here,  $\mathbf{x}_i$  is the origin of NeRF $_i$ , and  $\gamma$  is a hyperparameter that controls the sharpness of the blend.

Let  $\bar{p}_{i,k}$  and  $\bar{\mathbf{c}}_{i,k}$  be the termination probability and color for sample  $k$



Figure 2.6: Qualitative comparison of NeRF blending methods. While individual NeRFs and baseline blending methods (IDW-2D, IDW-3D) struggle to reconstruct both chairs clearly, our proposed **IDW-Sample** method produces a sharp, high-quality rendering that successfully integrates the best information from both sources.

from  $\text{NeRF}_i$ . The final color for pixel  $j$ ,  $I^{(j)}$ , is then rendered as:

$$I^{(j)} = \sum_k \sum_i w_{i,k} \bar{p}_{i,k} \bar{c}_{i,k}. \quad (2.7)$$

The weights  $w_{i,k}$  are normalized in a two-step process: first, they are normalized across NeRFs for each sample ( $\sum_i w_{i,k} = 1, \forall k$ ), and then the entire volumetric sum is normalized to ensure the final pixel color is valid ( $\sum_k \sum_i w_{i,k} \bar{p}_{i,k} = 1$ ). This ensures that our blending respects the relative opacity contributions of samples along the ray while properly combining the photometric information from multiple sources. The result, as shown qualitatively in Figure 2.6, is a seamlessly blended image that often exhibits higher quality than any of the individual source NeRFs and other baselines.

## 2.4 Experiments

To validate the effectiveness of the NeRFuser framework, we conducted a comprehensive set of experiments, evaluating both the registration and blending components of our pipeline. This section details our experimental setup, the datasets used, and the quantitative and qualitative results of our evalua-

tions. While our framework is agnostic to the underlying NeRF implementation, all experiments utilize NeRFacto [119] as the base model, with default training hyperparameters. To ensure a fair evaluation of the registration performance, we disabled camera optimization during NeRF training.

### 2.4.1 Datasets

We evaluated our method on three distinct datasets, covering environments from small, object-centric indoor scenes to large-scale outdoor street views.

#### Object-Centric Indoor Scenes

To test our framework in controlled, real-world indoor environments, we collected a new dataset consisting of three scenes.<sup>2</sup> For each scene, we captured three video sequences using a handheld camera. Two sequences focus on two different, yet overlapping, objects of interest, which were used to train the individual NeRFs (NeRF A and NeRF B). A third, wider sequence capturing the entire scene was reserved for the test set. This dataset allows us to rigorously evaluate both the registration accuracy and the blending quality.

#### ScanNet Dataset

To assess the robustness and scalability of our registration method on a larger scale, we utilized the ScanNet dataset [8]. We selected the first 218 scenes and downsampled the frames to approximately 200 posed RGB-D images per scene. For each scene, these images were partitioned into three sets: two for training a pair of NeRFs and a third for testing. This large-scale benchmark enables a thorough comparison of our registration-from-re-rendering technique against traditional point-cloud-based registration baselines.

#### Mission Bay Dataset

To specifically evaluate the performance of our blending algorithm in a challenging outdoor environment, we used the Mission Bay dataset, originally introduced for Block-NeRF [77]. This dataset features an expansive street

---

<sup>2</sup>The dataset is publicly available at <https://huggingface.co/datasets/RIPL/TTIC-common/tree/main>.

| Blending             | Ground-truth ${}^A\mathbf{T}_B$ |                 |                    | Estimated ${}^A\hat{\mathbf{T}}_B$ |                 |                    |
|----------------------|---------------------------------|-----------------|--------------------|------------------------------------|-----------------|--------------------|
|                      | PSNR $\uparrow$                 | SSIM $\uparrow$ | LPIPS $\downarrow$ | PSNR $\uparrow$                    | SSIM $\uparrow$ | LPIPS $\downarrow$ |
| NeRF                 | 20.92                           | 0.716           | 0.369              | 20.90                              | 0.714           | 0.370              |
| Nearest (Block-NeRF) | 23.81                           | 0.779           | 0.283              | 23.68                              | 0.774           | 0.287              |
| IDW-2D (Block-NeRF)  | 24.70                           | 0.795           | 0.267              | 24.64                              | 0.792           | 0.267              |
| IDW-3D (Block-NeRF)  | 23.48                           | 0.776           | 0.279              | 23.45                              | 0.772           | 0.280              |
| IDW-Sample (Ours)    | <b>24.91</b>                    | <b>0.813</b>    | <b>0.228</b>       | <b>24.83</b>                       | <b>0.810</b>    | <b>0.229</b>       |

Table 2.1: Blending results on Object-Centric Indoor Scenes. *IDW-Sample* works the best for all metrics with both ground-truth and estimated transformations. Results with estimated  ${}^A\hat{\mathbf{T}}_B$  are only marginally worse than those with ground-truth  ${}^A\mathbf{T}_B$ , which demonstrates that our proposed NeRF registration is accurate enough for the downstream blending task.

scene and provides an ideal testbed for comparing our sample-based blending strategy against other state-of-the-art methods under conditions where ground-truth transformations between NeRFs are known.

### 2.4.2 Evaluation of NeRF Registration and Blending

We first present a holistic evaluation of the entire NeRFuser pipeline on our custom Object-Centric Indoor Scenes dataset. This experiment assesses the combined performance of our registration and blending techniques.

For the registration stage, we re-rendered 32 images for each NeRF from poses sampled on the upper hemisphere of a unit sphere (with elevation angles from  $0^\circ$  to  $30^\circ$ ). For the blending stage, we set the proximity test ratio to  $\tau = 1.8$  and the blending rate to  $\gamma = 5$ . Table 2.1 reports the quantitative results, averaged over all test images from the three scenes. The results demonstrate that our full pipeline (NeRFuser) consistently outperforms individual NeRFs and baseline blending methods across all metrics.

### 2.4.3 Quantitative Analysis of Registration

To isolate and rigorously evaluate the performance of our registration-from-re-rendering method, we conducted experiments on the ScanNet dataset. We followed the same registration procedure as before, but increased the number of re-rendered poses to 60 to handle the larger scene complexity.

| Registration | Point-cloud    | $\text{err}_r$ ( $^\circ$ ) | $\text{err}_t$ | $\text{err}_s$ | Success     |
|--------------|----------------|-----------------------------|----------------|----------------|-------------|
| ICP [1]      | NeRF-extracted | 3.027                       | 0.1151         | N/A            | 0.13        |
| FGR [7]      | NeRF-extracted | 4.549                       | 0.1844         | N/A            | 0.04        |
| FPFH [3]     | NeRF-extracted | 2.805                       | 0.0381         | N/A            | 0.17        |
| ICP [1]      | RGB-D-fused    | 1.598                       | 0.0816         | N/A            | 0.17        |
| FGR [7]      | RGB-D-fused    | 1.330                       | 0.0372         | N/A            | 0.71        |
| FPFH [3]     | RGB-D-fused    | <b>0.049</b>                | <b>0.0205</b>  | N/A            | <i>0.79</i> |
| NeRFuser     | N/A            | <i>0.588</i>                | <i>0.0315</i>  | <b>0.0211</b>  | <b>0.84</b> |

Table 2.2: ScanNet registration results. We compare against point-cloud registration methods on both NeRF-extracted and ground-truth RGB-D-fused point-clouds. Because NeRF reconstructions contain noisy geometry, registration on NeRF-extracted point-clouds underperforms. On RGB-D-fused point-clouds, NeRFuser matches baseline performance in  $\text{err}_r$  and  $\text{err}_t$  while achieving the highest success rate. Unlike point-cloud baselines, our method also recovers relative scale. Best and second-best results are shown in **bold** and *italics*, respectively.

We compare our method against several point-cloud registration (PCR) baselines. These baselines were applied to point-clouds extracted directly from the trained NeRFs, as well as higher-quality point-clouds fused from the ground-truth posed RGB-D images. For evaluating registration accuracy, we measure the rotational error ( $\text{err}_r$ ), translational error ( $\text{err}_t$ ), and scale error, where scale error is defined as  $\text{err}_s = |\log \Delta s|$ , with  $\Delta s$  being the ratio of the ground-truth scale to the estimated scale.

A registration trial was considered a failure if it produced NaN values or if the errors exceeded predefined thresholds:  $\text{err}_r > 5^\circ$ ,  $\text{err}_t > 0.2$ , or  $\text{err}_s > 0.1$ . Table 2.2 presents the results, averaged over all successfully registered scenes, alongside the overall success rate. Our method achieves a significantly higher success rate and lower error compared to the PCR baselines, highlighting the robustness of leveraging synthesized images for alignment.

#### 2.4.4 Quantitative Analysis of Blending

To specifically assess the quality of our proposed sample-based blending (IDW-Sample), we conducted experiments on the Mission Bay dataset, assuming ground-truth transformations were provided. This allows us to de-

| Blending   | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ |
|------------|-----------------|-----------------|--------------------|
| NeRF       | 17.306          | 0.571           | 0.502              |
| Nearest    | 19.070          | 0.657           | 0.398              |
| IDW-2D     | 19.692          | 0.659           | 0.413              |
| IDW-3D     | 18.806          | 0.636           | 0.433              |
| IDW-Sample | <b>19.986</b>   | <b>0.678</b>    | <b>0.388</b>       |

Table 2.3: Blending results on Mission Bay dataset.

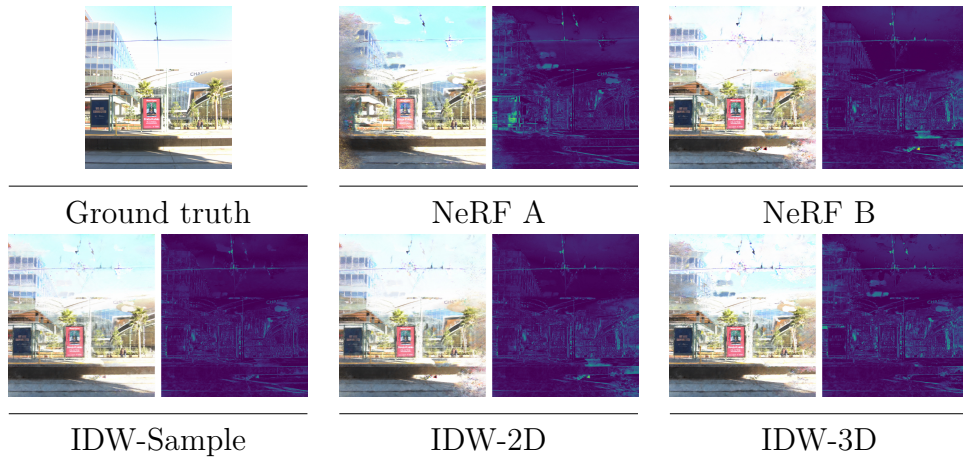


Figure 2.7: Qualitative comparison of NeRF blending methods on the Mission Bay dataset. Per-pixel errors are visualized as heat maps. Individual NeRF renderings exhibit large artifacts on either side of the view, which are best resolved by our proposed *IDW-Sample* blending technique.

couple the evaluation from any potential registration errors. For these experiments, we set the proximity test ratio to  $\tau = 1.2$  and the blending rate to  $\gamma = 10$ .

Table 2.3 reports the quantitative rendering metrics, averaged over all test images. Our IDW-Sample method achieves the best performance across all metrics, indicating its superior ability to fuse information from multiple NeRFs. Figure 2.7 provides a qualitative comparison, visualizing the per-pixel error as a heat map. The visualization clearly shows that individual NeRFs contain significant artifacts, which are most effectively resolved by our IDW-Sample approach, resulting in a more complete and coherent rendering.



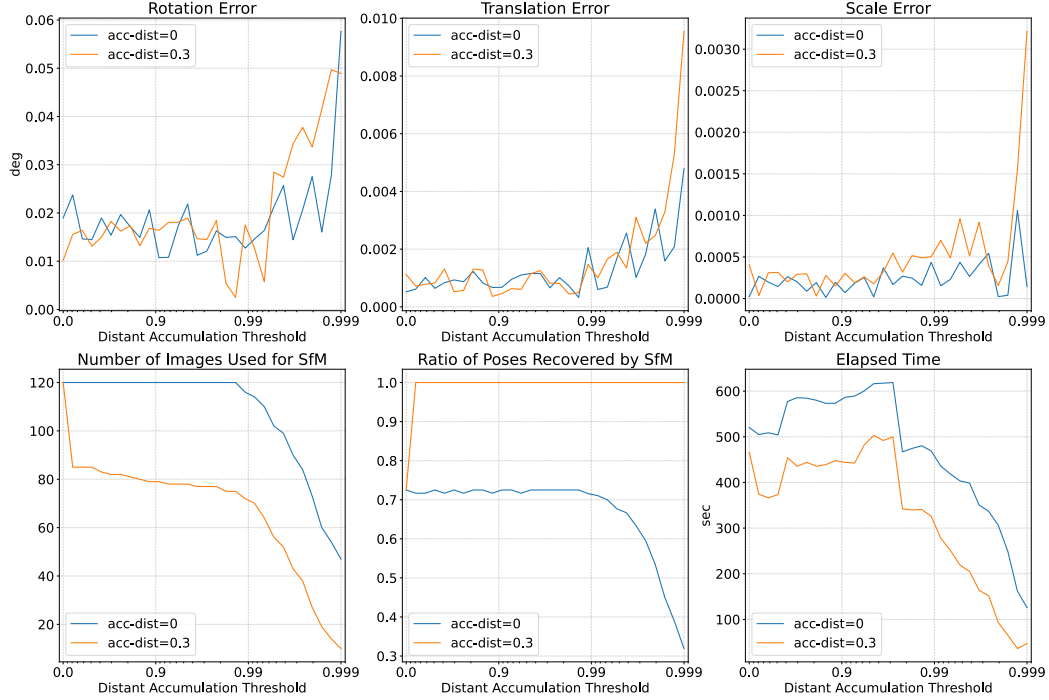


Figure 2.8: NeRF registration error and time consumption for different accumulation-based pose filtering thresholds with distant accumulation  $q_d$  ( $d = 0.3$ ) and regular accumulation ( $d = 0.0$ ).

### 2.4.5 Ablation Studies

We conducted several ablation studies to analyze the impact of key design choices and hyperparameters within the NeRFuser framework.

**Impact of Distant Accumulation for Pose Filtering.** We investigated the effectiveness of our proposed distant accumulation metric for filtering poor-quality re-rendered images prior to registration. Figure 2.8 shows the registration error and processing time for different filtering thresholds using both distant accumulation ( $d = 0.3$ ) and regular accumulation ( $d = 0.0$ ). The results show that our distant accumulation metric effectively filters out poor-quality images, leading to a significant reduction in registration time while maintaining comparable accuracy.



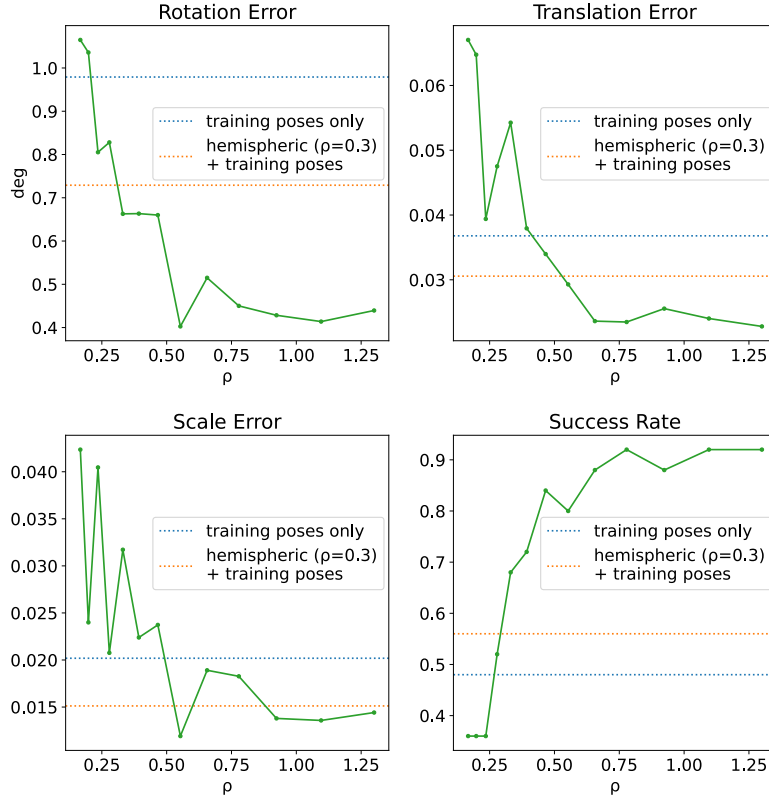


Figure 2.9: The effect of re-rendering poses on NeRF registration. The results show that increasing the number of sampled poses improves accuracy and success rate.

**Impact of Re-rendering Poses for Registration.** We studied how the number and type of poses used for re-rendering affect registration performance on the ScanNet dataset. We define  $\rho$  as the ratio of the number of sampled poses to the number of original training views and vary it geometrically. As shown in Figure 2.9, with more sampled poses, the registration errors go down while the success rate improves (green curve). Using additional hemispheric poses besides the training ones also proves helpful (orange line vs. blue line). More interestingly, with a large enough ratio  $\rho$ , registration with hemispherically sampled poses outperforms using the training poses, even when using the same number or fewer poses in total. This shows that it is beneficial to have a larger spatial span of re-rendering poses for registration, as illustrated in Figure 2.2.

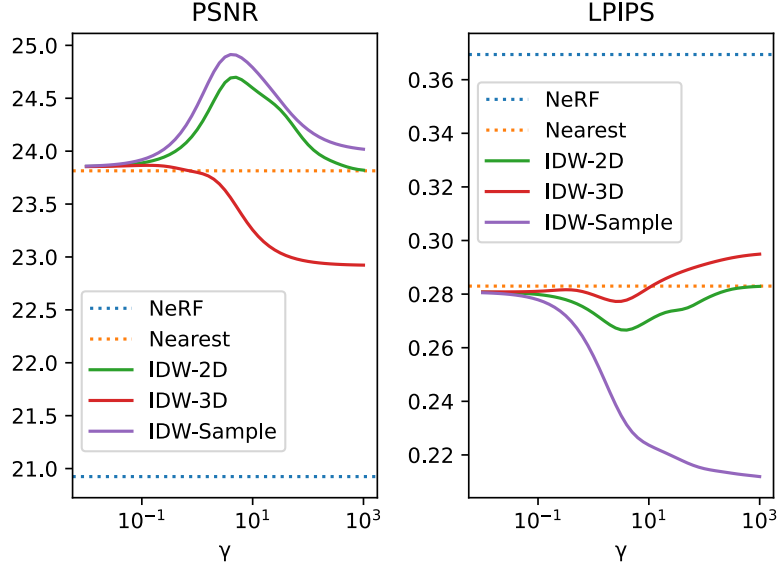


Figure 2.10: The effect of the blending rate  $\gamma$  on the performance of different IDW-based blending methods. Our IDW-Sample approach consistently achieves the highest PSNR across a wide range of  $\gamma$  values.

**Impact of Blending Rate  $\gamma$ .** Finally, we analyzed the effect of the blending rate hyperparameter,  $\gamma$ , on the performance of IDW-based blending methods. Using our Object-Centric Indoor Scenes dataset with ground-truth transformations, we varied  $\gamma$  across several orders of magnitude. Figure 2.10 plots the rendering quality (PSNR) for each blending method. The results show that for all IDW-based methods, there is an optimal range for  $\gamma$ . As  $\gamma \rightarrow 0$ , all methods converge to a simple average, while as  $\gamma \rightarrow \infty$ , they approach a nearest-neighbor selection. Critically, our proposed IDW-Sample method consistently outperforms the other baselines across nearly the entire range of  $\gamma$  values.

**Impact of Query Pose on Blending Performance.** To provide a more intuitive, geometric understanding of where our sample-based blending offers the most significant advantages, we conducted a qualitative ablation study on the effect of the query pose. Figure 2.11 visualizes the performance improvement of our *IDW-Sample* method over a *Nearest* baseline across a range of different camera viewpoints relative to the centers of the two source NeRFs. The results indicate that the greatest benefits are realized from poses

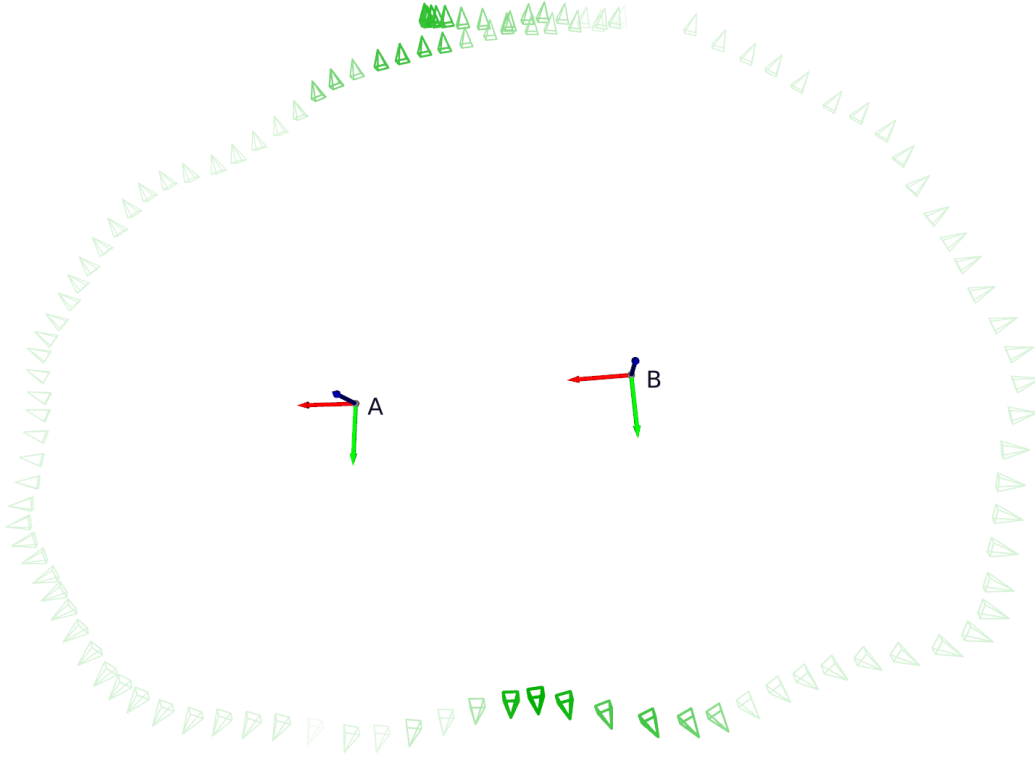


Figure 2.11: A visualization of the relative performance of *IDW-Sample* to the *Nearest* baseline. The axes represent the coordinate frames of two input NeRFs, and the green frusta denote test camera poses. The darker the frustum, the greater the performance gain of *IDW-Sample*. The best-performing poses are those that evenly observe the scene content from both NeRFs.

that are positioned to observe both NeRFs’ high-confidence regions evenly, particularly in the overlapping volume between them.

## 2.5 Implications for 3D Gaussian Splatting

More recently, 3D Gaussian Splatting (3DGS) [104] has emerged as a powerful explicit representation for 3D scenes, employing a collection of 3D Gaussian primitives to model geometry and appearance. While 3DGS offers significant advantages in rendering speed compared to NeRF—achieving real-time frame rates through rasterization rather than volumetric ray marching—the

fundamental challenges of representing large-scale environments remain. The problems of registering and blending multiple independently captured 3DGS models are directly analogous to the problems we address for NeRFs. Below we discuss how the two core components of NeRFuser could be adapted to the 3DGS setting, and where the explicit nature of Gaussians creates both opportunities and new challenges.

**Registration.** Our *registration from re-rendering* technique is largely representation-agnostic: its only requirement is the ability to synthesize images from novel viewpoints. Since 3DGS models are equally capable of rendering images, the same pipeline—sample poses on the upper hemisphere, render images from each model, run SfM on the combined image set—transfers with minimal modification. The distant accumulation metric (see Figure 2.3) would require adaptation, since 3DGS renders via alpha-compositing sorted Gaussians rather than integrating continuous density along a ray. However, a direct discrete analog exists: in 3DGS, one can alpha-composite all Gaussians in the standard depth-sorted order—so that near Gaussians properly attenuate those behind them—but sum the resulting rendering weights only for Gaussians whose depth exceeds the threshold  $d$ . This preserves exactly the same two failure-mode sensitivities: under-trained regions where opacity is spread thinly, and too-close cameras where all significant Gaussians lie closer than  $d$ .

That said, the explicit nature of 3DGS also opens a complementary registration pathway: the Gaussian centers themselves form a dense, colored point cloud that can be fed directly into traditional point cloud registration methods [1, 3, 7]. In fact, GaussReg [132] pursues exactly this strategy. However, as our ScanNet experiments suggest (Table 2.2), purely geometric registration can be brittle in the presence of noise or incomplete coverage. A hybrid strategy that first uses the explicit point cloud for a coarse initial alignment and then refines with an SfM-based pass on re-rendered images—combining the speed of explicit geometry with the robustness of photometric matching—would be a promising direction to explore.

**Blending.** Adapting the blending component is less straightforward. Our *sample-based blending* algorithm operates on volumetric ray samples: for each sample point along a ray, we compute an IDW weight and merge the density and color contributions from multiple NeRFs. This is natural for

NeRFs because rendering *is* ray marching. 3DGS, by contrast, renders via tile-based rasterization: Gaussians are projected to 2D, sorted by depth, and alpha-composited front-to-back, with no explicit set of ray samples to merge.

The most natural adaptation is *primitive-level fusion*: after registration, one simply collects the Gaussian primitives from all source models into a single scene, modulating each Gaussian’s opacity by an IDW factor based on the distance from its center to its source model’s origin—analogous to our per-sample weighting. This stays fully within the rasterization pipeline and preserves real-time rendering. More broadly, because Gaussians are discrete entities with well-defined spatial positions, weighting and merging them is a straightforward geometric operation, whereas NeRFs require the more involved volumetric sample merging that NeRFuser introduces. In this sense, the explicit nature of 3DGS actually simplifies the blending problem relative to NeRFs, even as it complicates the direct application of our specific ray-sample-level algorithm.

## 2.6 Relation to Subsequent Work

Since NeRFuser was first released, the dominant trend in large-scale 3D scene representation has shifted decisively toward explicit Gaussian methods. VastGaussian [148], Scaffold-GS [151], CityGaussian [150], and the hierarchical 3D Gaussian representation of Kerbl et al. [145] scale explicit primitives to city-block and larger scenes while retaining real-time rendering, and STORM [179] extends similar ideas to dynamic outdoor environments. The natural question this raises for NeRFuser—how its registration and blending algorithms would port to a 3DGS substrate—is the subject of Section 2.5.

On the narrower problem of registering and composing independently-trained scene representations, subsequent work has moved in two complementary directions. GaussReg [132] is the closest direct descendant on the explicit side, exploiting the Gaussian point cloud for a fast coarse-to-fine registration, and complements rather than replaces NeRFuser’s re-rendering-based approach (see Section 2.5 for a hybrid proposal). Adjacent hybrid and fused representations—NICER-SLAM [165], RadSplat [174], and NeuralGS [176]—sidestep the multi-model composition problem by fusing implicit and explicit fields into a single representation or by folding the scene into a unified SLAM-style pipeline.

For single-session, centrally-trained large-scale reconstruction, modern

3DGS methods now deliver better speed/quality tradeoffs than a NeRF composition would. NeRFuser’s continuing value lies in a more specific regime: first, the methodology of *registration from re-rendering*, which remains the right primitive whenever one has access only to pre-trained scene representations and not to their training images or poses; and second, the conceptual stance that scene representations should be treated as modularly composable objects—a property that matters most for decentralized, multi-agent, and lifelong mapping scenarios, and one that unified SLAM-style pipelines do not set out to provide.

## 2.7 Conclusion

This chapter introduced NeRFuser, a framework that directly confronts the critical challenge of spatial scalability in 3D scene reconstruction for robotic agents. We have demonstrated that by treating individual, pre-trained Neural Radiance Fields as modular building blocks, it is possible to construct a cohesive, high-fidelity representation of a large-scale environment. This is achieved without the need for the original training images or camera poses, a practical constraint that mirrors real-world, decentralized mapping scenarios where data may be captured by multiple agents or at different times.

The core contributions of this work are twofold. First, we proposed a novel *registration from re-rendering* technique, which leverages the view synthesis capability of NeRFs themselves to robustly solve for the full Sim(3) transformation between them. Complemented by our *distant accumulation* metric for filtering poor-quality renderings, this method provides a robust alternative to traditional registration techniques that rely on explicit geometric extraction. Second, we introduced a fine-grained, *sample-based blending* algorithm that intelligently merges volumetric information from multiple NeRFs at the ray-sample level. Our experiments have shown that this approach produces seamless, high-quality renderings that often surpass the quality of any single source NeRF, particularly in regions of overlap or sparse coverage.

By successfully registering and blending disparate NeRFs, NeRFuser provides the foundational component of this thesis: a practical and effective method for building the scalable 3D scene representation required for a robot to possess true spatial awareness. This static, comprehensive environmental map is an essential prerequisite for a robot’s broader understanding. However, a static representation is insufficient for meaningful interaction. A robot

must also comprehend the dynamic nature of its surroundings, particularly how objects can move and be manipulated.

The next chapter will build directly upon this foundation of scalable scene representation. We will shift our focus from the static geometry of the environment to the dynamic properties of the objects within it. Specifically, we will introduce SPLART, a method for reconstructing articulated objects and inferring their kinematic properties from sparse temporal observations. Having established *how* to build a large-scale map of the world, we will now explore *how* to understand the functional, movable objects that populate it, bringing us one step closer to enabling sophisticated robotic manipulation.

# Chapter 3

## Temporally-Aware Scene Understanding with SplArt

### 3.1 Introduction

In the preceding chapter, we addressed the fundamental challenge of spatial scalability in 3D scene reconstruction. We introduced NeRFuser, a framework for building comprehensive, high-fidelity models of large-scale, static environments by registering and blending multiple Neural Radiance Fields. This capability provides a robot with a crucial, detailed map of its surroundings—the foundation for spatial awareness. However, a purely static representation of the world is inherently incomplete. For a robot to transition from a passive observer to an active participant in its environment, it must understand not only *what* objects are and *where* they are, but also *how* they can move and be manipulated through purposeful interaction.

The majority of objects central to human activity are not rigid; they are articulated. Doors swing on hinges, drawers slide in and out, laptops fold, and scissors pivot. The ability to perceive and model this articulation is a cornerstone of embodied intelligence, as it unlocks the potential for meaningful physical interaction. Without an understanding of an object’s kinematics, a robot cannot reason about its functionality or plan effective manipulation strategies. For instance, to open a cabinet, a robot must recognize that the door is a distinct, movable part and infer the rotational motion it affords. This leap from static geometry to dynamic potential is the next critical step in building an actionable world model for robotics.



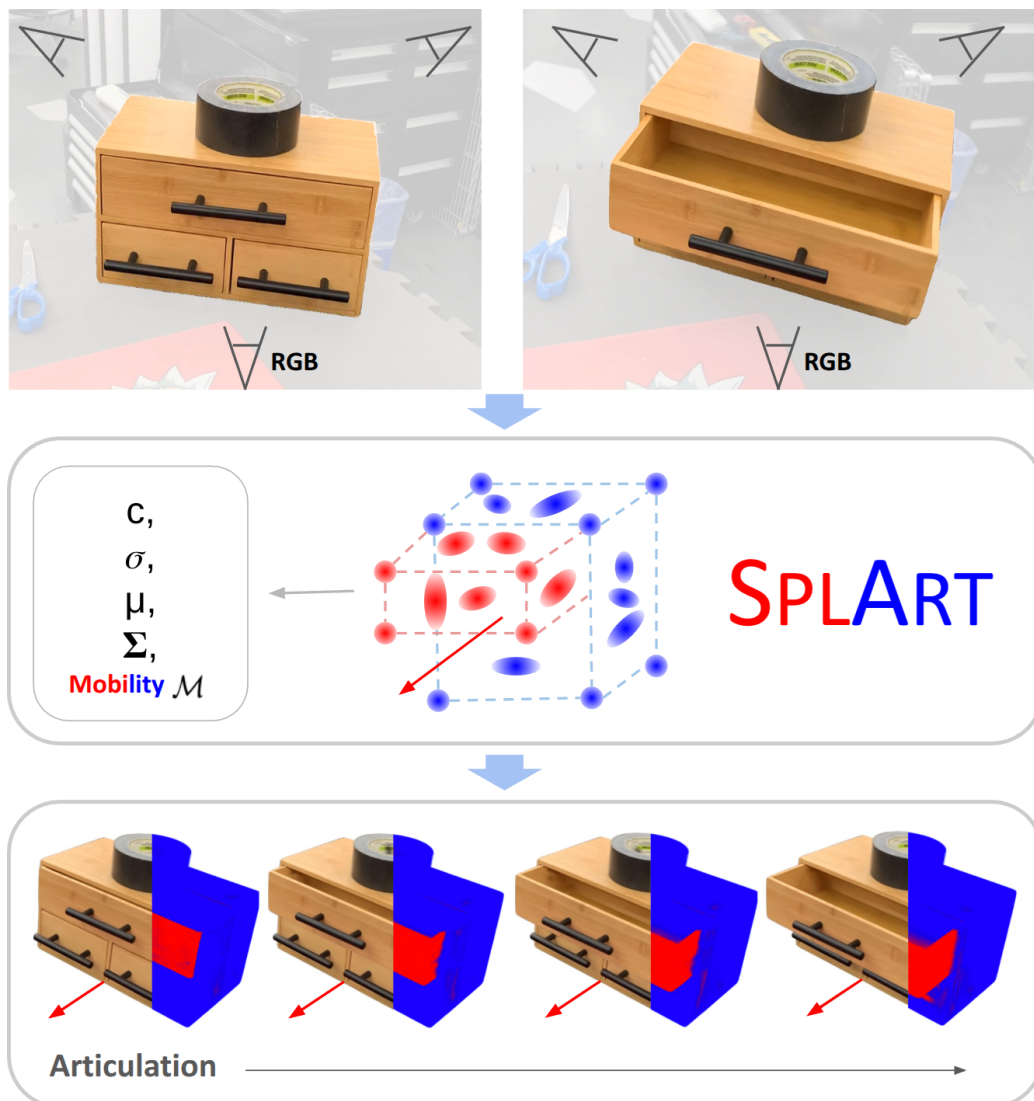


Figure 3.1: Given (top) RGB images of an object at two articulation states, (middle) SPLART uses 3DGS to simultaneously reconstruct its static and dynamic parts and estimate the kinematic articulation model. SPLART is then able to (bottom) render high-fidelity 3D reconstructions of the object along with part-level segmentations for novel articulation states, allowing for novel view synthesis.

This chapter introduces **SplArt**,<sup>1</sup> a novel framework that directly confronts this challenge by reconstructing articulated objects and inferring their kinematic properties from sparse temporal data [171] (see Figure 3.1). Traditional approaches to this problem often suffer from significant limitations: they may require extensive 3D supervision, costly manual annotations [28, 40, 66], or category-specific prior knowledge [22], hindering their scalability and practical deployment. Furthermore, many existing methods do not support the real-time, photorealistic rendering necessary for dynamic simulation and closed-loop control.

SPLART overcomes these hurdles by adopting a self-supervised, category-agnostic approach. We demonstrate that by observing an articulated object in just two distinct end states—for example, a cabinet door open and closed—our method can robustly solve for its underlying articulation model. Central to our approach is the use of 3D Gaussian Splatting (3DGS) [104], an explicit scene representation that enables both high-fidelity reconstruction and real-time rendering ( $> 100$  FPS). We augment the standard 3DGS framework by associating each Gaussian primitive with a differentiable mobility parameter, which allows us to effectively segment the object into its static and mobile parts through optimization.

To ensure robust convergence, SPLART employs a multi-stage optimization strategy that decouples the complex problem into three manageable steps: initial per-state reconstruction, mobility estimation for part segmentation, and finally, joint refinement of the articulation parameters. This structured process, guided by geometric self-supervision losses, allows SPLART to accurately estimate the articulation model (be it revolute or prismatic) without requiring any 3D ground truth or category-specific templates.

By enabling the reconstruction of dynamic, articulated objects, SPLART provides the second major component of this thesis. Building upon the spatially-scalable scene representation from NeRFuser, we now enrich this model with a temporal and functional understanding of the objects within it. Having established *how* to build a large-scale map and *how* to understand the movable objects it contains, we will have laid the necessary perceptual groundwork to, in the final chapter, empower a robotic agent with the high-level reasoning needed to understand human language and perform complex, multi-step tasks.

---

<sup>1</sup>Code is publicly available at <https://github.com/ripl/splart>.

## 3.2 Related Work

The core challenge addressed in this chapter—reconstructing articulated objects from temporally-sparse visual data—builds upon decades of research in 3D computer vision, robotics, and machine learning. Our work is positioned at the intersection of three key areas: methods for learning object articulation from 3D data, the evolution of 3D representations for reconstruction, and the recent emergence of self-supervised techniques that leverage differentiable rendering.

### 3.2.1 Data-Driven Articulation Learning

Estimating the kinematic properties of articulated objects is a long-standing problem, essential for robotic interaction and manipulation [69, 93, 94, 143]. Early approaches often relied on predefined models or significant human intervention. More recently, the field has shifted toward data-driven methods that learn articulation properties directly from sensor data, most commonly 3D point-clouds.

A significant body of work has focused on training deep neural networks end-to-end to infer articulation parameters. For instance, Shape2Motion [22] analyzes motion parts from a single point-cloud, while other methods perform category-level pose estimation at the cost of requiring class-specific models [10, 13, 28]. To improve generalization across object types, methods like RPM-Net [23] and OPD [65] were developed to predict part segmentation and kinematics without being tied to a specific category. Similarly, DITTO [66] learns to predict motion and geometry from pairs of 3D point-clouds without explicit labels. Other works have explored inferring articulation from different modalities or for specific downstream tasks [35, 98, 109].

A common thread among these powerful learning-based methods is their significant reliance on large quantities of labeled training data, often in the form of 3D ground truth with part-level segmentations and kinematic annotations [28, 66]. This requirement for costly 3D supervision fundamentally limits their scalability and practicality in real-world robotic scenarios where such annotations are unavailable. In contrast, our work, SPLART, aims to reconstruct both the detailed geometry and appearance while inferring articulation in a self-supervised manner, obviating the need for 3D supervision or category-specific priors.

### 3.2.2 Representations for 3D Object Reconstruction

The choice of 3D representation is fundamental to any reconstruction task. Early methods in 3D reconstruction often predicted explicit representations such as point-clouds [9, 62], voxels [4], or meshes. While direct and interpretable, these representations can be memory-intensive or struggle to capture fine geometric detail and photorealistic appearance.

The advent of implicit neural representations marked a significant paradigm shift. Methods like DeepSDF [18] and Occupancy Networks [16] learn to represent geometry as a continuous function, enabling the reconstruction of detailed and watertight surfaces [63, 163]. Building on this, Neural Radiance Fields (NeRFs) [39], as discussed in the previous chapter, extended this concept to model both geometry and appearance, enabling photorealistic view synthesis through differentiable volumetric rendering. Several surface-based rendering techniques have also leveraged this capability [42, 46, 47].

However, the computational cost of NeRF’s volumetric rendering process makes it too slow for many real-time robotics applications. More recently, 3D Gaussian Splatting (3DGS) [104] has emerged as a powerful alternative, representing scenes with an explicit collection of 3D Gaussian primitives. This formulation, combined with a differentiable rasterization pipeline, allows for both high-fidelity reconstruction and rendering at real-time rates ( $> 100$  FPS), a crucial advantage for dynamic simulation and visual feedback in robotics. Other recent works, such as SUGAR [141], have also explored hybrid representations to achieve similar goals.

Beyond rendering speed, the explicit, primitive-based nature of 3DGS offers a methodological advantage that is particularly relevant for articulated object reconstruction. Because each Gaussian is a discrete entity with well-defined spatial attributes, it can be individually parameterized—for instance, with a per-primitive mobility label that distinguishes static from mobile parts—and directly subjected to geometric operations such as rigid transformations and nearest-neighbor distance computations. These per-primitive operations, which are central to SPLART’s optimization strategy (Section 3.4), have no straightforward analog in NeRF’s continuous density field, which encodes the scene monolithically in network weights rather than as a collection of manipulable primitives. As discussed in Section 2.5, this explicitness simplifies the manipulation and fusion of scene primitives more broadly, further motivating the transition from the implicit representations of the previous chapter.

### 3.2.3 Articulated Object Reconstruction

The power of differentiable 3D representations has recently been harnessed to jointly reconstruct articulated objects and infer their kinematic parameters. Many of these approaches are training-based, using large synthetic 3D datasets to train models that can predict joint parameters and segment parts for novel objects [40, 66, 67, 74, 78, 98, 99].

To reduce the dependency on large, annotated datasets, a new class of self-supervised methods has emerged. These approaches optimize a scene representation on a per-instance basis, using photometric and geometric consistency as supervision. PARIS [108] and Articulated-NeRF [168] use NeRF-based representations to reconstruct objects from images captured in different states. Other works have incorporated depth data [157], explored different consistency losses [109], or focused on robotic interaction [146, 156]. Some recent methods have also begun to tackle objects with more complex, multi-part articulations, though they often require the number of parts to be known a priori and assume a simple kinematic chain [157, 168, 172]. Concurrently, some works are exploring the use of large language or vision-language models to bring semantic priors to the articulation estimation problem [170, 173].

The rapid adoption of 3D Gaussian Splatting has led to a surge of interest in articulated 3DGS. Recent works have also pushed the boundaries of self-supervised articulation learning with Gaussians. ArtGS [172] and REArtGS [178] explore similar directions, often with different constraints or supervision signals. ArticulatedGS [169], in particular, achieves robust digital twin modeling without 3D supervision. In the wild, ArtiPoint [177] extends these capabilities to dynamic videos using point tracking. Furthermore, Articulate-Anything [170] demonstrates the potential of Vision-Language Models to automate this process across diverse categories, highlighting the growing intersection between 3D reconstruction and semantic understanding.

Our method, SPLART, contributes to this line of self-supervised research. It is the first to apply the highly efficient 3DGS representation to this task without requiring any 3D supervision or pre-trained, category-specific priors. By introducing a novel multi-stage optimization strategy and geometric self-supervision, it robustly handles challenging cases and achieves state-of-the-art performance while enabling real-time rendering, making it a practical solution for robotic applications.

### 3.3 Preliminaries

3D Gaussian Splatting (3DGS) [104] is a contemporary method for reconstructing 3D scenes from posed images. It represents the scene as a collection of 3D Gaussian distributions in a continuous space. Each Gaussian primitive is parameterized by its mean (position)  $\mu \in \mathbb{R}^3$ , a rotation matrix  $\mathbf{R} \in \mathbb{R}^{3 \times 3}$  (orientation), a scaling matrix  $\mathbf{S} \in \mathbb{R}^{3 \times 3}$ , and an opacity value  $\sigma \in \mathbb{R}^+$ . The rotation and scaling matrices together define the covariance of the Gaussian.

The influence of a single Gaussian on any point  $\mathbf{x}$  in 3D space is defined by its probability density function:

$$g(\mathbf{x}|\mu, \mathbf{R}, \mathbf{S}, \sigma) = \sigma \exp \left( -\frac{1}{2}(\mathbf{x} - \mu)^\top (\mathbf{R}\mathbf{S}\mathbf{S}^\top \mathbf{R}^\top)^{-1} (\mathbf{x} - \mu) \right). \quad (3.1)$$

For rendering, these 3D Gaussians are projected into 2D camera space. The Gaussians that intersect a given pixel’s viewing ray are then sorted by depth and alpha-composited to produce the final pixel color. This explicit representation and efficient rasterization pipeline allow 3DGS to achieve significantly faster rendering speeds (often over  $100\times$ ) compared to implicit representations like Neural Radiance Fields (NeRFs) [39], making it highly suitable for real-time applications. For more technical details, we refer the reader to the original 3DGS paper [104] and the `gsplat` library [161].

### 3.4 Methodology

The core objective of SPLART is to reconstruct an articulated object at the part level and simultaneously estimate its kinematic motion model. This is achieved using only two sets of posed RGB images, each capturing the object in one of its two distinct end states. We assume the articulation type—either revolute (rotational) or prismatic (translational)—is known beforehand. Our approach is built upon an extension of 3D Gaussian Splatting (3DGS) and employs a carefully designed multi-stage optimization process to ensure robust and accurate convergence without requiring 3D supervision.

#### 3.4.1 Problem Formulation

Let us consider an object composed of a static *parent* part and a movable *child* part. The input to our system consists of two sets of observations,

$\mathcal{O}_l = \{(I_l^i, \mathbf{P}_l^i, \mathbf{K}_l^i)\}_{i=1}^{N_l}$ , for the two end states, where  $l \in \{0, 1\}$ . Here,  $I_l^i$  is an RGB image, and  $\mathbf{P}_l^i$  and  $\mathbf{K}_l^i$  are the corresponding camera extrinsics and intrinsics, all defined within a common world coordinate frame.

A revolute articulation is modeled by a rotation axis  $\mathbf{a}$  ( $\|\mathbf{a}\| = 1$ ), a pivot point  $\mathbf{p}$  on that axis, and a rotation angle  $\theta$ . A point  $\mathbf{x}$  on the mobile part at state  $l = 0$  is transformed to its position at state  $l = 1$  by the rigid transformation  ${}^1\mathbf{T}_0$ :

$${}^1\mathbf{T}_0(\mathbf{x}) = \mathbf{R}_{\mathbf{a},\theta}(\mathbf{x} - \mathbf{p}) + \mathbf{p}, \quad (3.2)$$

where  $\mathbf{R}_{\mathbf{a},\theta}$  is the rotation matrix derived from the axis-angle representation. A prismatic articulation is modeled by a translation axis  $\mathbf{a}$  ( $\|\mathbf{a}\| = 1$ ) and a translation distance  $d$ . The goal is to reconstruct the object's geometry and appearance at the part level while estimating the transformation  ${}^1\mathbf{T}_0$ .

### 3.4.2 Articulation Operator for Gaussian Primitives

Having defined how a 3D point transforms under articulation, we now lift the same rigid transformation to the Gaussian primitives that make up our representation. Specifically, a Gaussian is: (1) rotated by an angle  $\theta$  around an axis defined by a pivot point  $\mathbf{p}$  and a direction vector  $\mathbf{a}$ ; and (2) translated along the same axis by a distance  $d$ . This transformation requires updating the Gaussian's parameters  $(\mu', \mathbf{R}', \mathbf{S}', \sigma')$  to reflect its new state. Furthermore, since the radiance field of the Gaussian (represented by Spherical Harmonics) is defined in its local frame and does not change, we must also compute the transformed view direction  $\mathbf{d}'$  for correctly querying the color.

Let  $\mathbf{x}'$  be the new position of a point  $\mathbf{x}$  after applying the articulated motion. The transformation is given by:

$$\mathbf{x}' = \mathbf{R}_{\mathbf{a},\theta}(\mathbf{x} - \mathbf{p}) + \mathbf{p} + d\mathbf{a}, \quad (3.3)$$

where  $\mathbf{R}_{\mathbf{a},\theta}$  is the rotation matrix corresponding to the axis-angle pair  $(\mathbf{a}, \theta)$ . By definition, the influence of the transformed Gaussian at the new point  $\mathbf{x}'$  must equal the influence of the original Gaussian at the original point  $\mathbf{x}$ :

$$g(\mathbf{x}' \mid \mu', \mathbf{R}', \mathbf{S}', \sigma') = g(\mathbf{x} \mid \mu, \mathbf{R}, \mathbf{S}, \sigma) \quad \forall \mathbf{x}. \quad (3.4)$$

By substituting Equations (3.1) and (3.3) into Equation (3.4), we can derive

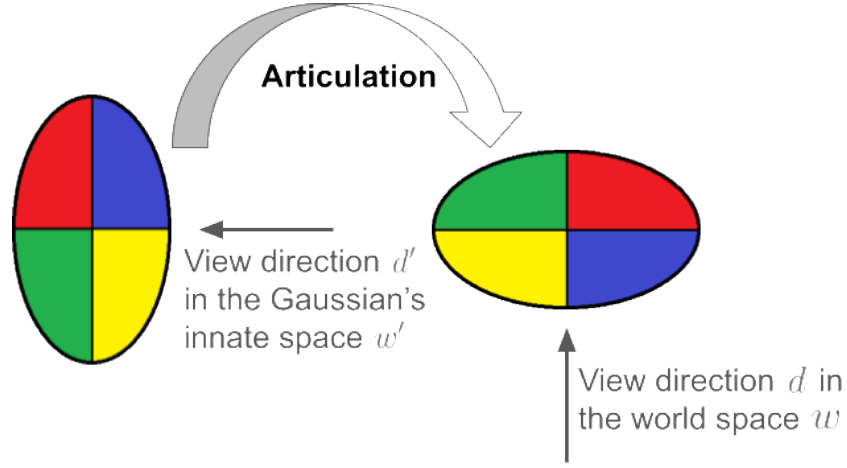


Figure 3.2: To compensate for the articulation of the Gaussian primitive, the view direction is transformed into the Gaussian’s local coordinate frame before querying the radiance field, which itself remains unchanged.

the updated parameters for the articulated Gaussian:

$$\mu' = \mathbf{R}_{a,\theta}(\mu - \mathbf{p}) + \mathbf{p} + d\mathbf{a}, \quad (3.5a)$$

$$\mathbf{R}' = \mathbf{R}_{a,\theta}\mathbf{R}, \quad (3.5b)$$

$$\mathbf{S}' = \mathbf{S}, \quad (3.5c)$$

$$\sigma' = \sigma. \quad (3.5d)$$

To determine the correct view direction  $\mathbf{d}'$  for querying the radiance field, we consider a local coordinate frame  $w'$  that moves with the Gaussian. A point  $\mathbf{x}'$  in the world frame  $w$  corresponds to the point  $\mathbf{x}$  in the local frame  $w'$ . The transformation from the local frame to the world frame is  ${}^w\mathbf{T}_{w'}(\mathbf{x}) = \mathbf{R}_{a,\theta}(\mathbf{x} - \mathbf{p}) + \mathbf{p} + d\mathbf{a}$ . Therefore, a view direction  $\mathbf{d}$  in the world frame must be transformed into the local frame for the color query:

$$\mathbf{d}' = {}^{w'}\mathbf{T}_w(\mathbf{d}) \quad (3.6a)$$

$$= \mathbf{R}_{a,\theta}^{-1}\mathbf{d}. \quad (3.6b)$$

Figure 3.2 provides a conceptual illustration of this process.



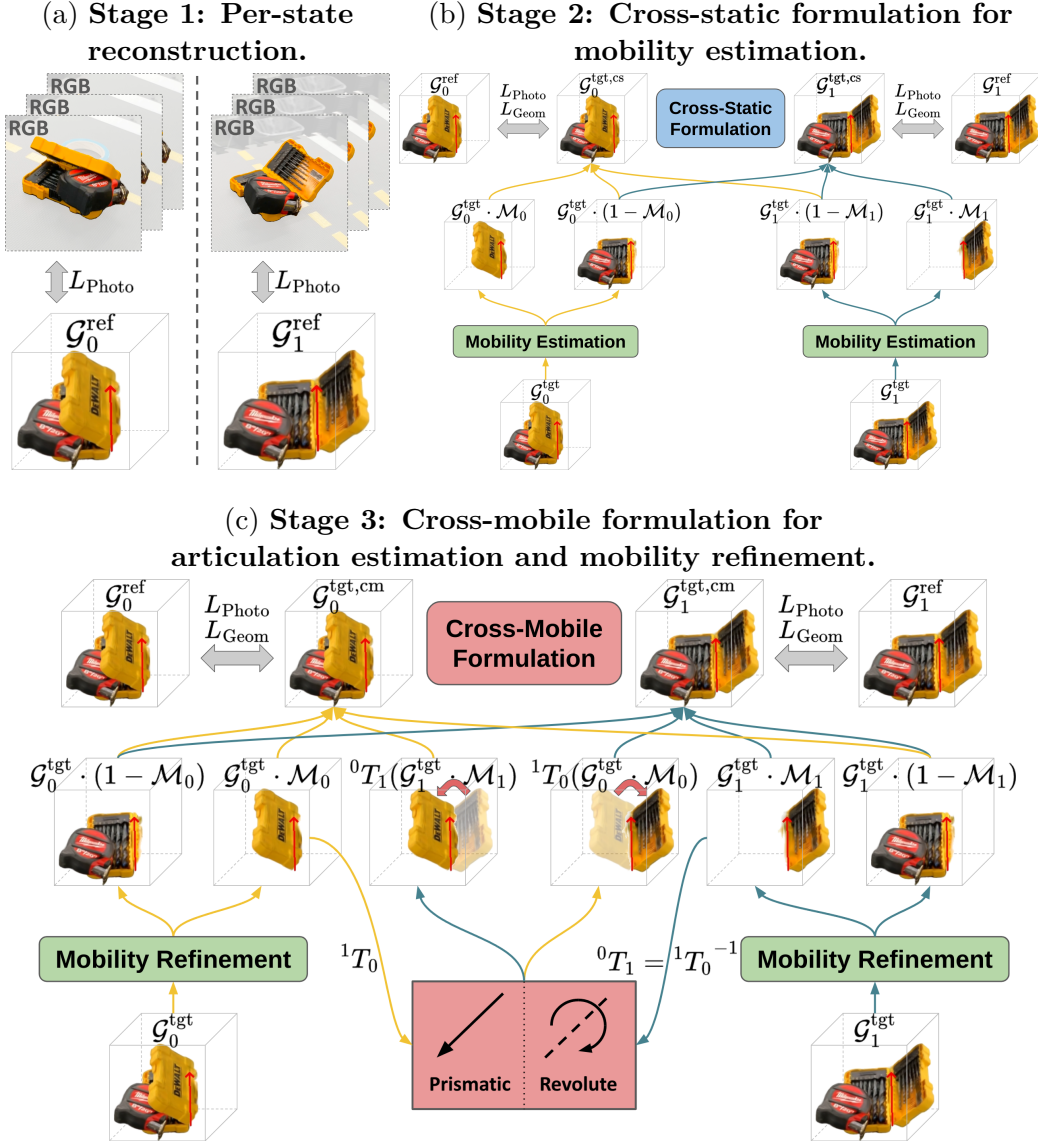


Figure 3.3: **Overview of the SplArt methodology.** Our approach consists of three decoupled stages to ensure stable optimization. (a) Stage 1 constructs an independent 3DGS model for each state using a standard photometric loss. (b) Stage 2 introduces a cross-static formulation to estimate the mobility of each Gaussian, effectively segmenting the object into static and mobile parts. (c) Stage 3 employs a cross-mobile formulation to estimate the articulation parameters and jointly refine the part segmentation.

### 3.4.3 Decoupled Multi-Stage Optimization

Jointly optimizing for geometry, appearance, part segmentation, and articulation parameters from scratch is highly susceptible to converging to poor local minima [108, 168]. To ensure stable and accurate convergence, we decouple the part-level reconstruction and articulation estimation into a three-stage process, as illustrated in Figure 3.3.

#### Stage 1: Independent Per-State Reconstruction

The first stage focuses solely on reconstructing the object’s appearance and geometry at each of the two observed states independently. We optimize two separate 3DGS models, one for each state  $l \in \{0, 1\}$ , using the standard 3DGS training procedure. Each Gaussian in these initial models is assigned a persistent state label,  $l$ . We denote the resulting set of Gaussians for state  $l$  as  $\mathcal{G}_l^{\text{ref}}$ , where the superscript emphasizes that this is a reference reconstruction. The optimization for each state is guided by a photometric loss  $\Delta_I$ :

$$\min_{\mathcal{G}_l^{\text{ref}}} \Delta_I(\hat{I}_l^{\text{ref},i}, I_l^i), \quad (3.7a)$$

$$\hat{I}_l^{\text{ref},i} = R(\mathcal{G}_l^{\text{ref}}, \mathbf{P}_l^i, \mathbf{K}_l^i), \quad (3.7b)$$

where  $R(\cdot)$  is the differentiable 3DGS rendering function. This stage provides two high-fidelity, but independent, representations of the object.

#### Stage 2: Mobility Estimation via Cross-Static Formulation

The second stage aims to segment the object into its static and mobile parts by estimating a *mobility* value for each Gaussian primitive. To begin, we duplicate the reference Gaussians  $\mathcal{G}_l^{\text{ref}}$  to create target representations  $\mathcal{G}_l^{\text{tgt}}$  for each state. These target Gaussians will be modified in subsequent stages, while the reference models remain fixed.

We extend each Gaussian in  $\mathcal{G}^{\text{tgt}}$  with a differentiable mobility parameter  $m \in [0, 1]$ , initialized to 0.5. This parameter allows us to decompose any Gaussian into a static component and a mobile component. The opacity  $\sigma$  of the original Gaussian is split between these two components: the static component receives an opacity of  $\sigma \cdot (1 - m)$ , and the mobile component receives  $\sigma \cdot m$ . We denote the set of all mobilities as  $\mathcal{M}$ .

The key insight for this stage is that the static parts of the object should be geometrically consistent across both states. We formalize this with a

**cross-static (cs) formulation**, where the object’s complete static geometry  $\mathcal{G}^s$  is represented by the union of the static components from both states:

$$\mathcal{G}^s = \mathcal{G}_l^{\text{tgt}} \cdot (1 - \mathcal{M}_l) \oplus \mathcal{G}_{1-l}^{\text{tgt}} \cdot (1 - \mathcal{M}_{1-l}), \quad (3.8)$$

where  $\oplus$  denotes concatenation. The full representation for rendering state  $l$  is then the combination of this shared static part and its own mobile part:

$$\mathcal{G}_l^{\text{tgt,cs}} = \mathcal{G}^s \oplus (\mathcal{G}_l^{\text{tgt}} \cdot \mathcal{M}_l). \quad (3.9)$$

This stage proceeds in two sub-stages:

**Stage 2a: Coarse Mobility Estimation with Geometric Consistency.**

We first obtain a coarse estimate of the mobilities using geometric supervision alone. To measure the geometric distance between two sets of Gaussians, we introduce a weighted version of the Chamfer distance. Specifically, letting  $\mathcal{X} = \{(\mathbf{x}_i, v_i)\}_{i=1}^M$  and  $\mathcal{Y} = \{(\mathbf{y}_j, w_j)\}_{j=1}^N$  be two sets of point-weight pairs, the weighted Chamfer distance is defined as:

$$\begin{aligned} \text{Chamfer}(\mathcal{X}, \mathcal{Y}) = & \sum_{(\mathbf{x}_i, v_i) \in \mathcal{X}} \tilde{v}_i \min_{(\mathbf{y}_j, w_j) \in \mathcal{Y}} \|\mathbf{x}_i - \mathbf{y}_j\|^2 \\ & + \sum_{(\mathbf{y}_j, w_j) \in \mathcal{Y}} \tilde{w}_j \min_{(\mathbf{x}_i, v_i) \in \mathcal{X}} \|\mathbf{x}_i - \mathbf{y}_j\|^2, \end{aligned} \quad (3.10)$$

where  $\tilde{v}_i = v_i / \sum v_i$  and  $\tilde{w}_j$  are the normalized weights. In our application, each Gaussian is treated as a point at its mean position, and its weight is its mobility-modified opacity.

The mobilities are then optimized by minimizing this Chamfer distance between the cross-static representation  $\mathcal{G}_l^{\text{tgt,cs}}$  and the original reference reconstruction  $\mathcal{G}_l^{\text{ref}}$  for both states:

$$\min_{\mathcal{M}} \text{CD}_0^{\text{cs}} + \text{CD}_1^{\text{cs}} + \lambda_{\text{geom}} \|\mathcal{M}\|, \quad (3.11a)$$

$$\text{where } \text{CD}_l^{\text{cs}} = \text{Chamfer}(\mathcal{G}_l^{\text{tgt,cs}}, \mathcal{G}_l^{\text{ref}}). \quad (3.11b)$$

A regularization term  $\lambda_{\text{geom}} \|\mathcal{M}\|$  is added to encourage smaller mobility values, preventing the trivial solution where all Gaussians are mobile ( $m = 1$ ). This step is computationally fast and provides a reasonable initialization for the next, more intensive refinement.

**Stage 2b: Joint Refinement with Photometric Supervision.** Next, we refine the mobilities and the full set of Gaussian attributes (position, rotation, scale, etc.) using photometric supervision. We optimize the target Gaussians  $\mathcal{G}^{\text{tgt}}$  and the mobilities  $\mathcal{M}$  by rendering the cross-static representation and minimizing the photometric loss against the input images:

$$\min_{\mathcal{G}^{\text{tgt}}, \mathcal{M}_{1-l}} \Delta_I(\hat{I}_l^{\text{cs}}, I_l) + \lambda_{\text{photo}} \|\mathcal{M}_{1-l}\|, \quad (3.12a)$$

$$\hat{I}_l^{\text{cs}} = \text{R}(\mathcal{G}_l^{\text{tgt,cs}}), \quad (3.12b)$$

where  $\lambda_{\text{photo}} \|\mathcal{M}_{1-l}\|$  is a regularization term similar to that in Equation (3.11a). This joint optimization produces a much more accurate part segmentation, preparing the model for the final stage of articulation estimation.

### Stage 3: Articulation Estimation via Cross-Mobile Formulation

The final stage estimates the articulation parameters that define the transformation between the two states. The core intuition here is that the mobile part in state 0, when transformed by the articulation motion, should align perfectly with the mobile part in state 1. We introduce a **cross-mobile (cm) formulation** to enforce this consistency. The mobile part at state  $l$ , denoted  $\mathcal{G}_l^{\text{m}}$ , is now represented as the union of its own mobile component and the *transformed* mobile component from the other state:

$$\mathcal{G}_l^{\text{m}} = (\mathcal{G}_l^{\text{tgt}} \cdot \mathcal{M}_l) \oplus {}^l\mathbf{T}_{1-l}(\mathcal{G}_{1-l}^{\text{tgt}} \cdot \mathcal{M}_{1-l}), \quad (3.13)$$

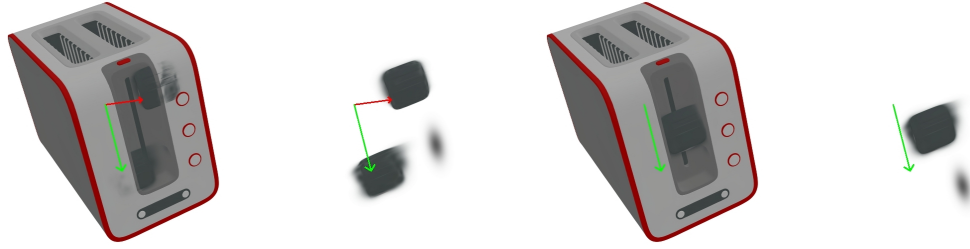
where  ${}^l\mathbf{T}_{1-l}$  is the rigid transformation from state  $1-l$  to  $l$ . The full representation for rendering state  $l$  thus becomes:

$$\mathcal{G}_l^{\text{tgt,cm}} = \mathcal{G}^{\text{s}} \oplus \mathcal{G}_l^{\text{m}}, \quad (3.14)$$

with  $\mathcal{G}^{\text{s}}$  defined as in Equation (3.8). This stage is also divided into sub-stages for robustness.

**Stage 3a: Coarse Articulation Estimation with Geometric Consistency.** Similar to Stage 2a, we coarsely estimate the articulation parameters  $\mathbf{T}$  by minimizing the geometric discrepancy between the cross-mobile representation and the reference model. This **Cross-Mobile Geometric Consistency** loss is defined as

$$\text{CD}_l^{\text{cm}} = \text{Chamfer}(\mathcal{G}_l^{\text{tgt,cm}}, \mathcal{G}_l^{\text{ref}}). \quad (3.15)$$



(a) Failure with cross-mobile geometric consistency.

(b) Success with mobile-only geometric consistency.

Figure 3.4: Handling geometrically ambiguous parts. For the toaster, the flat slider is geometrically similar to the static body. The standard cross-mobile consistency fails because nearest-neighbor associations in the Chamfer distance are error-prone. The mobile-only formulation, which focuses exclusively on aligning the mobile parts, successfully estimates the articulation.

However, we observe that this formulation is susceptible to local minima, particularly if the mobile part is geometrically insignificant compared to the static part, as illustrated in Figure 3.4.

To remedy this, we propose an alternative formulation, **Mobile-Only Geometric Consistency**, which directly aligns the mobile parts from the two states:

$$\text{CD}^{\text{m}} = \text{Chamfer}(\mathcal{G}_l^{\text{tgt}} \cdot \mathcal{M}_l, {}^l\mathbf{T}_{1-l}(\mathcal{G}_{1-l}^{\text{tgt}} \cdot \mathcal{M}_{1-l})). \quad (3.16)$$

While this successfully addresses scenarios like the one in Figure 3.4, the mobile-only formulation can fall short in other circumstances, especially when the visible portions of the mobile part differ significantly between the two states (Figure 3.5).

To take advantage of both formulations and to facilitate robustness in the inherently non-convex optimization of articulation parameters, we propose the following practical strategy:

1. With  $K^{\text{m}}$  randomized tries, we perform an initial articulation estimation via the mobile-only geometric consistency:

$$\mathbf{T}^{\text{m}} = \arg \min_{\mathbf{T}} \text{CD}^{\text{m}}. \quad (3.17)$$

2. With  $K^{\text{cm}}$  randomized tries, plus another try initialized with the best result from the previous step ( $\mathbf{T}^{\text{m}}$ ), we perform a second estimation via

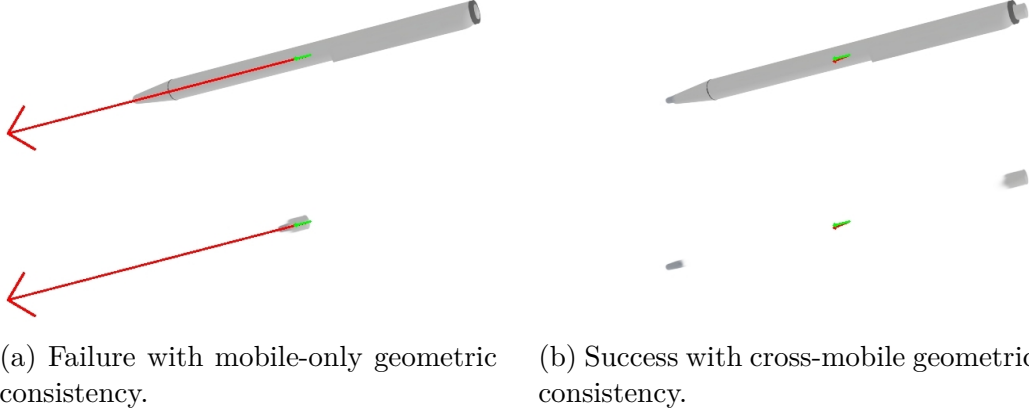


Figure 3.5: Handling partial observations. For the pen, only one end of the mobile part is visible in each state. The mobile-only consistency fails because the reconstructed mobile components are geometrically dissimilar. The cross-mobile consistency, which considers the joint representation, correctly estimates the articulation.

the cross-mobile geometric consistency:

$$\mathbf{T}^{\text{cm}} = \arg \min_{\mathbf{T}} (\text{CD}_0^{\text{cm}} + \text{CD}_1^{\text{cm}}). \quad (3.18)$$

3. A final run, initialized with the best result from the second step ( $\mathbf{T}^{\text{cm}}$ ), jointly refines both the articulation parameters and the mobilities by minimizing the cross-mobile geometric consistency loss:

$$\min_{\mathcal{M}, \mathbf{T}} (\text{CD}_0^{\text{cm}} + \text{CD}_1^{\text{cm}}). \quad (3.19)$$

This hierarchical approach of using multiple randomized trials and progressively refining the estimate significantly improves the likelihood of converging to the correct articulation model.

**Stage 3b: Joint Refinement with Photometric Supervision.** With a coarse estimate of the articulation parameters, we perform a joint optimization over all parameters—Gaussian attributes, mobilities, and articulation—

using the photometric rendering loss:

$$\min_{\mathcal{G}^{\text{tgt}}, \mathcal{M}_{1-l}, {}^l\mathbf{T}_{1-l}} \Delta_I(\hat{I}_l^{\text{cm}}, I_l), \quad (3.20a)$$

$$\hat{I}_l^{\text{cm}} = \text{R}(\mathcal{G}_l^{\text{tgt}, \text{cm}}). \quad (3.20b)$$

**Stage 3c: Mobility Correction.** Finally, we perform a quick geometric correction step on the mobilities. Since photometric supervision is limited to the training views, some Gaussians might be incorrectly assigned a high mobility if the articulation moves them out of view. A final optimization pass using only the cross-mobile geometric consistency loss, with fixed articulation parameters, corrects these errors.

### 3.4.4 Application to Real-World Data

To apply SPLART to objects in the real world, we establish a pipeline that begins with image capture from a handheld camera. For each of the two articulation states, we capture a set of images from various viewpoints around the object. We then use SAM 2 [175] to perform foreground-background segmentation, which separates the articulated object (the foreground) from the static background. We run COLMAP [5], a state-of-the-art Structure-from-Motion (SfM) tool enhanced with SuperPoint [12] descriptors and SuperGlue [31] matching, on the segmented background regions from *both* sets of images simultaneously. Since the background is static across both articulation states, this process robustly registers all camera poses into a single, consistent world coordinate frame. The segmented foreground images and their corresponding registered poses then provide the necessary high-quality input for the main SPLART optimization pipeline.

## 3.5 Experiments

To rigorously evaluate the performance of SPLART, we conducted a comprehensive suite of experiments designed to assess its capabilities in articulation estimation, part-level reconstruction, and real-world applicability. This section details the datasets used for evaluation, the metrics for quantifying performance, the baselines against which we compare, and a thorough analysis of the quantitative and qualitative results, including ablation studies that validate our core design choices.

### 3.5.1 Experimental Setup

#### Datasets

Our evaluation leverages both existing and newly curated datasets to ensure a comprehensive assessment across a diverse range of objects and scenarios.

**PARIS PartNet-Mobility Subset (PARIS-PMS).** We first evaluate on the dataset introduced by PARIS [108], which consists of 10 object instances (8 revolute, 2 prismatic) from the PartNet-Mobility dataset [32]. For each object, 100 rendered views are provided for training at each of the two end states, with 50 novel views reserved for testing. While the original dataset did not include ground-truth depth or segmentation maps, we augmented it by following the original data generation protocol to enable a more thorough evaluation. A key limitation of this dataset is that its test set only includes views of the two end states, precluding a quantitative evaluation of novel articulation synthesis.

**SplArt PartNet-Mobility Subset (SplArt-PMS).** To overcome the limitations of the PARIS-PMS dataset and to provide a more challenging benchmark, we curated a new, more comprehensive subset from PartNet-Mobility, which we refer to as SPLART-PMS. This dataset comprises 22 object instances from categories not present in PARIS-PMS, including 12 with revolute and 10 with prismatic articulations. Crucially, the test set for SPLART-PMS includes views of the object at intermediate articulation states, uniformly sampled from the range  $[-0.1, 1.1]$ . This enables a direct quantitative assessment of novel articulation synthesis and demands a higher degree of accuracy in all aspects of the reconstruction.

**Real-World Dataset.** To demonstrate the practical utility of SPLART in real-world settings, we collected a new dataset of 7 common household and office objects, featuring a total of 9 different articulations (e.g., monitor stand, cabinet door, eyeglasses). All images were captured using a standard handheld smartphone camera, and the camera poses were subsequently calibrated using the SfM-based pipeline detailed in Section 3.4.4. This dataset serves to validate the robustness of our full pipeline, from data capture to final reconstruction.



## Evaluation Metrics

We evaluate performance across two primary tasks: articulation estimation and part-level reconstruction.

**Articulation Estimation.** The accuracy of the estimated articulation model is measured based on its type.

- For a **revolute** articulation, defined by a pivot point  $\mathbf{p}$ , rotation axis  $\mathbf{a}$ , and angle  $\theta$ , we report three metrics: the angular error of the axis ( $\text{err}_a$ ), the geodesic distance between the predicted and ground-truth rotations ( $\text{err}_r$ ), and the pivot point error ( $\text{err}_p$ ), calculated as the closest distance between the predicted and ground-truth rotation axes.
- For a **prismatic** articulation, defined by a translation axis  $\mathbf{a}$  and distance  $d$ , we report the axis error ( $\text{err}_a$ ) and the Euclidean distance between the predicted and ground-truth translation vectors ( $\text{err}_t$ ).

**Part-Level Reconstruction.** Reconstruction quality is assessed from three perspectives: photometric accuracy, geometric fidelity, and segmentation correctness, all evaluated on novel test views.

- **Photometric Quality:** We measure the visual quality of rendered RGB images using the Peak Signal-to-Noise Ratio (PSNR).
- **Geometric Accuracy:** We evaluate geometric fidelity in two ways. First, we compute the Mean Absolute Error (MAE) between the rendered and ground-truth depth maps. Second, we extract meshes from the 3DGS representation and compute the Chamfer Distance (CD) between the reconstructed and ground-truth meshes for the static part ( $\text{CD}_s$ ), mobile part ( $\text{CD}_m$ ), and the whole object ( $\text{CD}_w$ ).
- **Part Segmentation Accuracy:** We measure the correctness of the part segmentation by rendering segmentation maps and calculating the mean Intersection-over-Union (mIoU) across the static part, mobile part, and background.

**Robustness and Success Criteria.** Recognizing that the joint optimization problem is susceptible to local minima, we conduct ten trials for each experiment with different random seeds to ensure a robust evaluation. A trial is deemed successful only if all of the following criteria are met:  $\text{err}_a < 5^\circ$ ;  $\text{err}_p < 0.05$  (for revolute);  $\text{err}_r < 10^\circ$  (for revolute); and  $\text{err}_t < 0.05$  (for prismatic). We report the success rate (number of successful runs out of ten) and present all other metrics averaged *only over the successful runs*.

### Baselines

We compare SPLART against two state-of-the-art methods:

- **PARIS** [108]: This method addresses the same task setup as SPLART, using a NeRF-based representation to jointly optimize for shape, appearance, and articulation from two sets of multi-view RGB images.
- **DTA** [157]: This method also reconstructs articulated objects from two states but requires RGB-D images as input. It is important to note that DTA produces object meshes that are not suitable for photorealistic rendering, unlike SPLART and PARIS.

### Implementation Details

The SPLART framework builds on two open-source libraries: `nerfstudio` [119] for its neural rendering pipeline and `gsplat` [161] for its Gaussian splatting implementation. We maintain consistent hyperparameters across all experiments, including both synthetic and real-world datasets.

The optimization stages have different termination criteria. Stages driven by geometric consistency (Stages 2a, 3a, and 3c) are run until convergence, which is determined by monitoring the rate of change of the loss function. In contrast, stages guided by a photometric loss (Stages 1, 2b, and 3b) are run for a fixed number of iterations: 10 000, 5000, and 10 000, respectively. For context, the default 3DGS implementation in `nerfstudio` (`splatfacto`) trains for 30 000 iterations.

In Stage 3a, we set the number of randomized trials to  $K^m = K^{\text{cm}} = 3$ . Each of these trials takes between 10 seconds and 1 minute to run on an NVIDIA RTX 2080 Ti GPU. The total training time for SPLART is approximately 20–30 minutes, which is about twice as long as a standard `splatfacto` run. For comparison, PARIS [108] requires 15–20 minutes, and DTA [157]

| Type      | Method           | Success<br>rate % $\uparrow$ | $\text{err}_a$<br>( $\times 10^{-2}$ DEG) $\downarrow$ | $\text{err}_p$<br>( $\times 10^{-3}$ ) $\downarrow$ | $\text{err}_r$<br>( $\times 10^{-2}$ DEG) $\downarrow$ | $\text{err}_t$<br>( $\times 10^{-3}$ ) $\downarrow$ | $\text{CD}_s$<br>( $\times 10^{-3}$ ) $\downarrow$ | $\text{CD}_m$<br>( $\times 10^{-3}$ ) $\downarrow$ | $\text{CD}_w$<br>( $\times 10^{-3}$ ) $\downarrow$ |
|-----------|------------------|------------------------------|--------------------------------------------------------|-----------------------------------------------------|--------------------------------------------------------|-----------------------------------------------------|----------------------------------------------------|----------------------------------------------------|----------------------------------------------------|
| Revolute  | PARIS            | 40.0                         | 121.90 $\pm$ 68.45                                     | 5.27 $\pm$ 4.55                                     | 127.80 $\pm$ 55.12                                     | N/A                                                 | 3.48 $\pm$ 0.39                                    | 68.44 $\pm$ 13.03                                  | 13.11 $\pm$ 1.16                                   |
|           | DTA <sup>†</sup> | 98.8                         | 12.32 $\pm$ 4.16                                       | 1.86 $\pm$ 1.69                                     | 19.41 $\pm$ 16.28                                      | N/A                                                 | 2.20 $\pm$ 0.49                                    | 1.27 $\pm$ 1.87                                    | 1.73 $\pm$ 0.03                                    |
|           | SPLART           | <b>100.0</b>                 | <b>3.70 <math>\pm</math> 0.38</b>                      | <b>0.40 <math>\pm</math> 0.07</b>                   | <b>4.85 <math>\pm</math> 0.33</b>                      | N/A                                                 | 4.08 $\pm$ 0.32                                    | <b>1.06 <math>\pm</math> 0.78</b>                  | 3.59 $\pm$ 0.21                                    |
| Prismatic | PARIS            | 50.0                         | 27.97 $\pm$ 13.09                                      | N/A                                                 | N/A                                                    | 4.28 $\pm$ 3.02                                     | 9.21 $\pm$ 1.94                                    | 151.78 $\pm$ 35.00                                 | 7.99 $\pm$ 0.49                                    |
|           | DTA <sup>†</sup> | 100.0                        | 16.26 $\pm$ 3.54                                       | N/A                                                 | N/A                                                    | 1.15 $\pm$ 0.14                                     | 2.69 $\pm$ 0.04                                    | 15.74 $\pm$ 0.28                                   | 2.22 $\pm$ 0.03                                    |
|           | SPLART           | 95.0                         | <b>2.36 <math>\pm</math> 0.44</b>                      | N/A                                                 | N/A                                                    | <b>0.33 <math>\pm</math> 0.04</b>                   | 6.02 $\pm$ 0.20                                    | 16.77 $\pm$ 1.14                                   | 3.69 $\pm$ 0.14                                    |
| Overall   | PARIS            | 42.0                         | 103.12 $\pm$ 57.38                                     | 5.27 $\pm$ 4.55                                     | 127.80 $\pm$ 55.12                                     | 4.28 $\pm$ 3.02                                     | 4.63 $\pm$ 0.70                                    | 85.11 $\pm$ 17.42                                  | 12.09 $\pm$ 1.03                                   |
|           | DTA <sup>†</sup> | 99.0                         | 13.11 $\pm$ 4.04                                       | 1.86 $\pm$ 1.69                                     | 19.41 $\pm$ 16.28                                      | 1.15 $\pm$ 0.14                                     | 2.30 $\pm$ 0.40                                    | 4.16 $\pm$ 1.55                                    | 1.83 $\pm$ 0.03                                    |
|           | SPLART           | <b>99.0</b>                  | <b>3.44 <math>\pm</math> 0.39</b>                      | <b>0.40 <math>\pm</math> 0.07</b>                   | <b>4.85 <math>\pm</math> 0.33</b>                      | <b>0.33 <math>\pm</math> 0.04</b>                   | 4.47 $\pm$ 0.30                                    | 4.20 $\pm$ 0.85                                    | 3.61 $\pm$ 0.19                                    |

Table 3.1: PARIS-PMS articulation and mesh reconstruction metrics. Best results in **bold**. <sup>†</sup>DTA requires ground-truth depth.

takes 35–45 minutes. At inference time, SPLART can render novel views and articulations at 60–100 frames per second, with performance varying based on scene complexity.

### 3.5.2 Quantitative and Qualitative Results

The main text reports scene-averaged metrics and representative qualitative comparisons; per-scene tables and additional qualitative results are given in Appendix A.

#### Articulation Estimation Accuracy

We first evaluate the accuracy of the estimated kinematic models. Tables 3.1 and 3.2 present the scene-averaged articulation errors on the PARIS-PMS and SPLART-PMS datasets, respectively. SPLART consistently and significantly outperforms the baselines across both datasets and for both articulation types. On the more challenging SPLART-PMS benchmark, our method achieves a much higher success rate, demonstrating the robustness of our multi-stage optimization strategy. The qualitative results in Figure 3.6 further illustrate this, where the estimated articulation axis and motion (red arrow) from SPLART closely match the ground truth (green arrow), while the baseline struggles.

#### Part-Level Reconstruction Quality

Next, we assess the quality of the part-level reconstructions. Tables 3.3 and 3.4 report the novel view synthesis metrics. SPLART achieves superior performance in photometric rendering (PSNR), geometric accuracy (Depth

| Type      | Method           | Success<br>rate % $\uparrow$ | $\text{err}_a$<br>( $\times 10^{-2}$ DEG) $\downarrow$ | $\text{err}_p$<br>( $\times 10^{-3}$ ) $\downarrow$ | $\text{err}_r$<br>( $\times 10^{-2}$ DEG) $\downarrow$ | $\text{err}_t$<br>( $\times 10^{-3}$ ) $\downarrow$ |
|-----------|------------------|------------------------------|--------------------------------------------------------|-----------------------------------------------------|--------------------------------------------------------|-----------------------------------------------------|
| Revolute  | PARIS            | 18.3                         | $167.34 \pm 42.47$                                     | $9.36 \pm 3.73$                                     | $270.37 \pm 91.19$                                     | N/A                                                 |
|           | DTA <sup>†</sup> | 33.3                         | $32.76 \pm 2.38$                                       | <b><math>1.28 \pm 0.39</math></b>                   | $40.80 \pm 4.66$                                       | N/A                                                 |
|           | SPLART-2a        | <b>78.5</b>                  | <b><math>2.91 \pm 0.55</math></b>                      | $2.53 \pm 2.18$                                     | $52.06 \pm 32.00$                                      | N/A                                                 |
|           | SPLART-2b        | 67.0                         | $4.62 \pm 1.69$                                        | $3.92 \pm 3.07$                                     | $131.99 \pm 64.45$                                     | N/A                                                 |
|           | SPLART-3a        | 0.8                          | $296.61 \pm 0.00$                                      | $2.83 \pm 0.00$                                     | $427.10 \pm 0.00$                                      | N/A                                                 |
|           | SPLART-3b        | 76.7                         | $84.13 \pm 19.13$                                      | $5.11 \pm 3.03$                                     | $206.98 \pm 56.30$                                     | N/A                                                 |
|           | SPLART           | 77.5                         | $20.76 \pm 0.81$                                       | $3.93 \pm 2.41$                                     | $78.31 \pm 31.06$                                      | N/A                                                 |
| Prismatic | PARIS            | 39.0                         | $49.49 \pm 18.92$                                      | N/A                                                 | N/A                                                    | $14.96 \pm 5.51$                                    |
|           | DTA <sup>†</sup> | 90.0                         | $102.77 \pm 12.48$                                     | N/A                                                 | N/A                                                    | $9.25 \pm 0.73$                                     |
|           | SPLART-2a        | 90.0                         | $25.88 \pm 23.25$                                      | N/A                                                 | N/A                                                    | $1.50 \pm 1.63$                                     |
|           | SPLART-2b        | 85.7                         | $15.63 \pm 20.40$                                      | N/A                                                 | N/A                                                    | <b><math>0.73 \pm 0.58</math></b>                   |
|           | SPLART-3a        | 13.0                         | <b><math>15.17 \pm 0.59</math></b>                     | N/A                                                 | N/A                                                    | $2.26 \pm 0.04$                                     |
|           | SPLART-3b        | 86.0                         | $61.00 \pm 19.76$                                      | N/A                                                 | N/A                                                    | $9.43 \pm 2.37$                                     |
|           | SPLART           | <b>95.0</b>                  | $25.72 \pm 17.63$                                      | N/A                                                 | N/A                                                    | $1.04 \pm 0.32$                                     |
| Overall   | PARIS            | 27.7                         | $103.88 \pm 29.79$                                     | $9.36 \pm 3.73$                                     | $270.37 \pm 91.19$                                     | $14.96 \pm 5.51$                                    |
|           | DTA <sup>†</sup> | 59.1                         | $77.76 \pm 8.87$                                       | <b><math>1.28 \pm 0.39</math></b>                   | $40.80 \pm 4.66$                                       | $9.25 \pm 0.73$                                     |
|           | SPLART-2a        | 83.7                         | $13.85 \pm 11.36$                                      | $2.53 \pm 2.18$                                     | $52.06 \pm 32.00$                                      | $1.50 \pm 1.63$                                     |
|           | SPLART-2b        | 74.7                         | <b><math>9.34 \pm 9.71</math></b>                      | $3.92 \pm 3.07$                                     | $131.99 \pm 64.45$                                     | <b><math>0.73 \pm 0.58</math></b>                   |
|           | SPLART-3a        | 6.4                          | $85.53 \pm 0.44$                                       | $2.83 \pm 0.00$                                     | $427.10 \pm 0.00$                                      | $2.26 \pm 0.04$                                     |
|           | SPLART-3b        | 80.9                         | $73.62 \pm 19.42$                                      | $5.11 \pm 3.03$                                     | $206.98 \pm 56.30$                                     | $9.43 \pm 2.37$                                     |
|           | SPLART           | <b>85.5</b>                  | $23.02 \pm 8.46$                                       | $3.93 \pm 2.41$                                     | $78.31 \pm 31.06$                                      | $1.04 \pm 0.32$                                     |

Table 3.2: SPLART-PMS articulation metrics. Best results in **bold**. <sup>†</sup>DTA requires ground-truth depth.

MAE), and part segmentation (mIoU). The mesh-based Chamfer Distance metrics, reported in the rightmost columns of Table 3.1, show that even without depth supervision, SPLART achieves geometric accuracy on par with DTA (which uses depth) and significantly better than PARIS.

Figure 3.6 provides a visual comparison of the reconstruction quality. The RGB renderings from SPLART are sharper and more detailed, and the part segmentation maps are cleaner and more accurate compared to the baseline. Furthermore, Figure 3.7 showcases one of SPLART’s key advantages: the ability to render high-fidelity, photorealistic views of the object at novel, intermediate articulation states, a capability not quantitatively evaluated by prior benchmarks but essential for robotics and simulation.

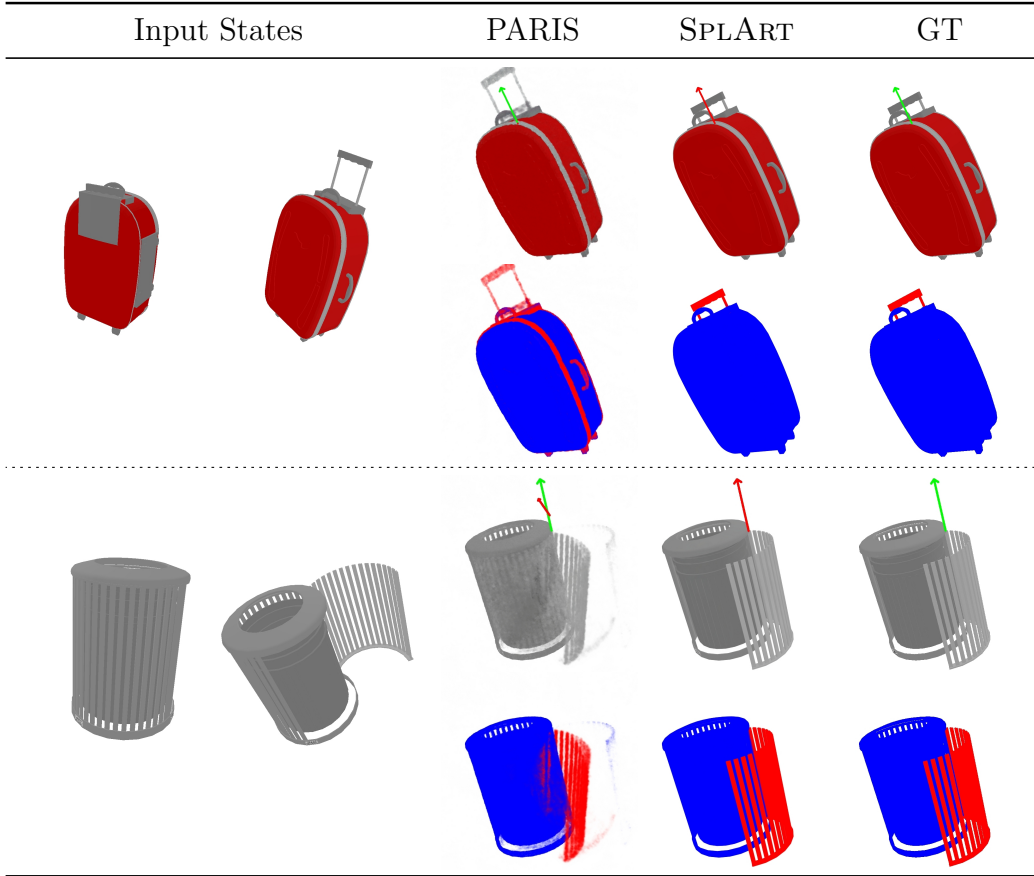


Figure 3.6: Qualitative comparison on part-level reconstruction and articulation estimation. For each object, the first row shows color renderings and the second row shows part segmentation maps. “GT” denotes ground truth.

### 3.5.3 Ablation Studies

To validate the contributions of the different components within our multi-stage optimization pipeline, we conducted a series of ablation studies on the more challenging SPLART-PMS dataset. We considered four variants of our method, each skipping a specific stage:

- **SplArt-2a:** Skips the coarse mobility estimation with geometric self-supervision (Stage 2a).
- **SplArt-2b:** Skips the mobility refinement with photometric supervi-

| Type      | Method           | PSNR $\uparrow$ | Depth MAE $\downarrow$ | mIoU $\uparrow$ |
|-----------|------------------|-----------------|------------------------|-----------------|
| Revolute  | PARIS            | 32.21           | 0.093                  | 0.955           |
|           | DTA <sup>†</sup> | N/A             | <b>0.031</b>           | 0.941           |
|           | SPLART           | <b>43.53</b>    | 0.039                  | <b>0.974</b>    |
| Prismatic | PARIS            | 33.75           | 0.108                  | <b>0.902</b>    |
|           | DTA <sup>†</sup> | N/A             | 0.066                  | 0.844           |
|           | SPLART           | <b>44.38</b>    | <b>0.025</b>           | 0.890           |
| Overall   | PARIS            | 32.52           | 0.096                  | 0.942           |
|           | DTA <sup>†</sup> | N/A             | 0.038                  | 0.922           |
|           | SPLART           | <b>43.70</b>    | <b>0.036</b>           | <b>0.957</b>    |

Table 3.3: PARIS-PMS novel view synthesis metrics, averaged over successful runs (see Table 3.1 for success rates). Best results in **bold**. <sup>†</sup>DTA requires ground-truth depth.

sion (Stage 2b).

- **SplArt-3a:** Skips the coarse articulation estimation with geometric self-supervision (Stage 3a).
- **SplArt-3b:** Skips the joint refinement of articulation with photometric supervision (Stage 3b).

The results, presented in Table 3.2, demonstrate that removing any of these stages leads to a degradation in performance, particularly in the success rate. This confirms that each stage plays a crucial role in stabilizing the optimization and guiding it toward the correct solution, highlighting the effectiveness of our decoupled, multi-stage design.

### 3.5.4 Robustness of Geometric Consistency

In Stage 3a, our strategy of using multiple randomized trials with both mobile-only and cross-mobile geometric consistency is key to achieving robust articulation estimation. For this strategy to be effective, two conditions must be met: (1) a correct articulation estimate should result in a lower loss value than most incorrect estimates; and (2) the optimization should have a high probability of converging to the global optimum within a small

| Type      | Method           | PSNR $\uparrow$ | Depth MAE $\downarrow$ | mIoU $\uparrow$ |
|-----------|------------------|-----------------|------------------------|-----------------|
| Revolute  | PARIS            | 32.75           | 0.125                  | 0.897           |
|           | DTA <sup>†</sup> | N/A             | 0.053                  | 0.886           |
|           | SPLART           | <b>36.38</b>    | <b>0.032</b>           | <b>0.897</b>    |
| Prismatic | PARIS            | 32.30           | 0.124                  | 0.902           |
|           | DTA <sup>†</sup> | N/A             | 0.060                  | 0.857           |
|           | SPLART           | <b>37.77</b>    | <b>0.033</b>           | <b>0.938</b>    |
| Overall   | PARIS            | 32.51           | 0.125                  | 0.900           |
|           | DTA <sup>†</sup> | N/A             | 0.058                  | 0.867           |
|           | SPLART           | <b>37.01</b>    | <b>0.032</b>           | <b>0.916</b>    |

Table 3.4: SPLART-PMS novel view and articulation synthesis metrics, averaged over successful runs (see Table 3.2 for success rates). Best results in **bold**. <sup>†</sup>DTA requires ground-truth depth.

number of randomized trials. To validate these conditions, we conducted an experiment with 200 independent optimization trials for several scenes from the SPLART-PMS dataset: 100 using mobile-only consistency and 100 using cross-mobile consistency. Figure 3.8 visualizes the results. The scatter plot shows the final loss rank for each trial, distinguishing between successful and failed runs. The results confirm our hypotheses: successful trials consistently cluster at the lower end of the loss spectrum, and the combination of both formulations ensures a high success rate across different object geometries.

### 3.5.5 Real-World Application

Finally, to demonstrate the practical viability of SPLART, we applied it to our custom dataset of real-world objects captured with a handheld camera. Figure 3.9 showcases the results. Despite the challenges of real-world data, including imperfect camera poses and lighting variations, SPLART successfully reconstructs a diverse set of articulated objects and accurately estimates their kinematic models. This validates our approach as a practical tool for building actionable, dynamic models of objects for robotic interaction.

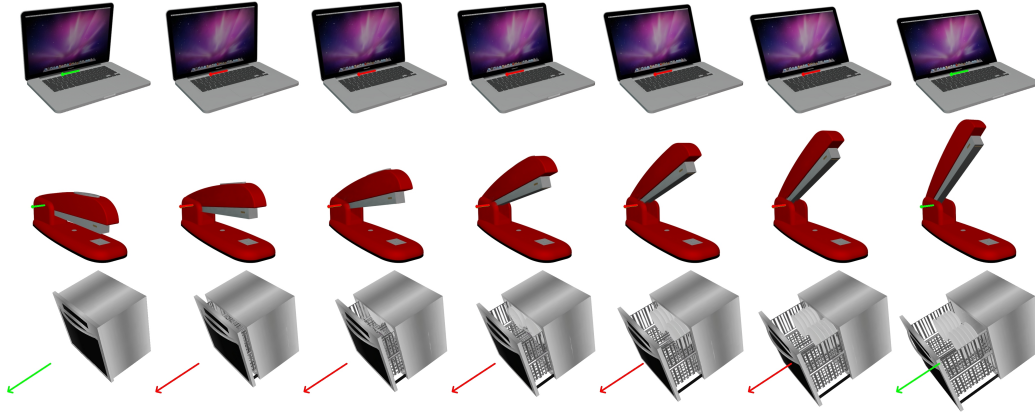


Figure 3.7: State interpolation from the estimated articulation.

## 3.6 Discussion

While SPLART provides a robust solution for two-part articulated objects, it is important to acknowledge its limitations and consider promising avenues for future research.

### 3.6.1 Scope and Limitations

The reliance on only two end-state observations, while powerful for data-sparse scenarios, can lead to inherent ambiguities. When multiple distinct articulation models can explain the observed change in state, the optimization may converge to a kinematically plausible but incorrect solution. As shown in Figure 3.10, this can manifest in scenarios such as a clock’s hand being modeled with an unexpected rotation axis or a globe’s surface spinning independently from its stand. These cases highlight the fundamental challenge of inferring motion from minimal temporal data and suggest that incorporating weak priors or a few intermediate observations could be a valuable avenue for future research.

### 3.6.2 Future Directions

A natural direction for future work is extending the framework to handle more complex, multi-part scenarios. SPLART is not fundamentally limited to two-part articulated objects. Assuming the number of DoFs is known, an



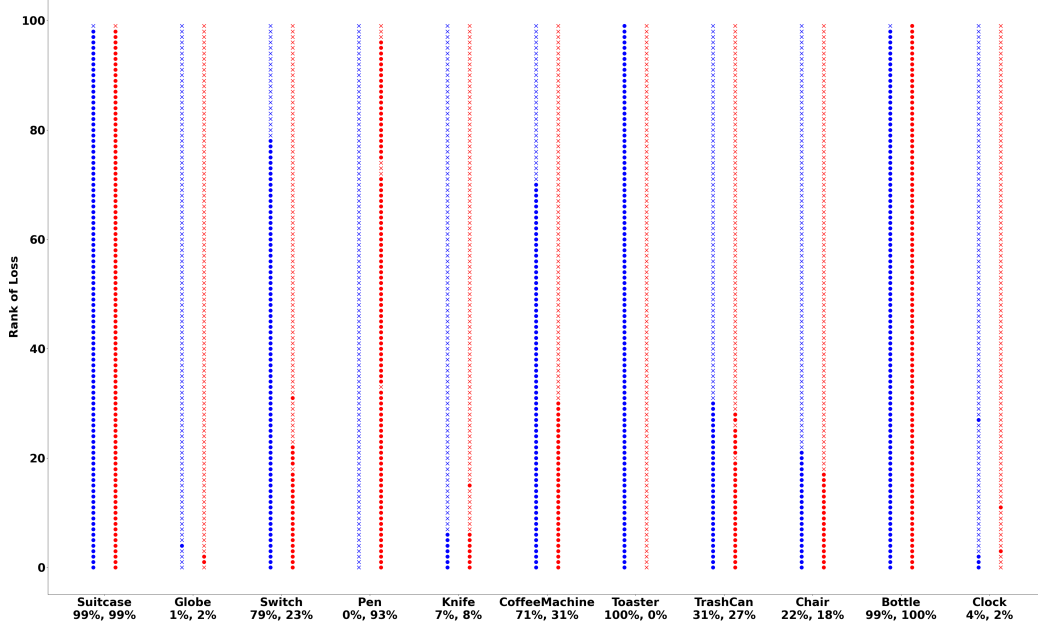


Figure 3.8: An analysis of the robustness of geometric consistency for articulation estimation. The horizontal axis represents different scenes, each with two columns for the two optimization formulations: **mobile-only** and **cross-mobile**. The vertical axis represents the loss rank (lower is better). For each trial,  $\bullet$  denotes success and  $\times$  denotes failure. The average success rates for each formulation are shown below the scene name. We observe that successful trials ( $\bullet$ ) are consistently associated with lower losses, and that both formulations are necessary to ensure overall robustness (e.g., the *pen* scene requires **cross-mobile** consistency, while the *toaster* scene requires **mobile-only** consistency).

extension would maintain the same initial two stages: reconstructing reference states and separating static from mobile components. The third stage would further entail identifying the specific part ID for each Gaussian across both states. This could involve associating each Gaussian with a part ID logit vector or adopting an MLP-based part assignment field, similar to [157]. Both of these approaches would yield a soft part ID assignment, thereby enabling optimization via our proposed self-supervised geometric consistency and photometric losses.

While feasible, this extension would increase the number of optimizable

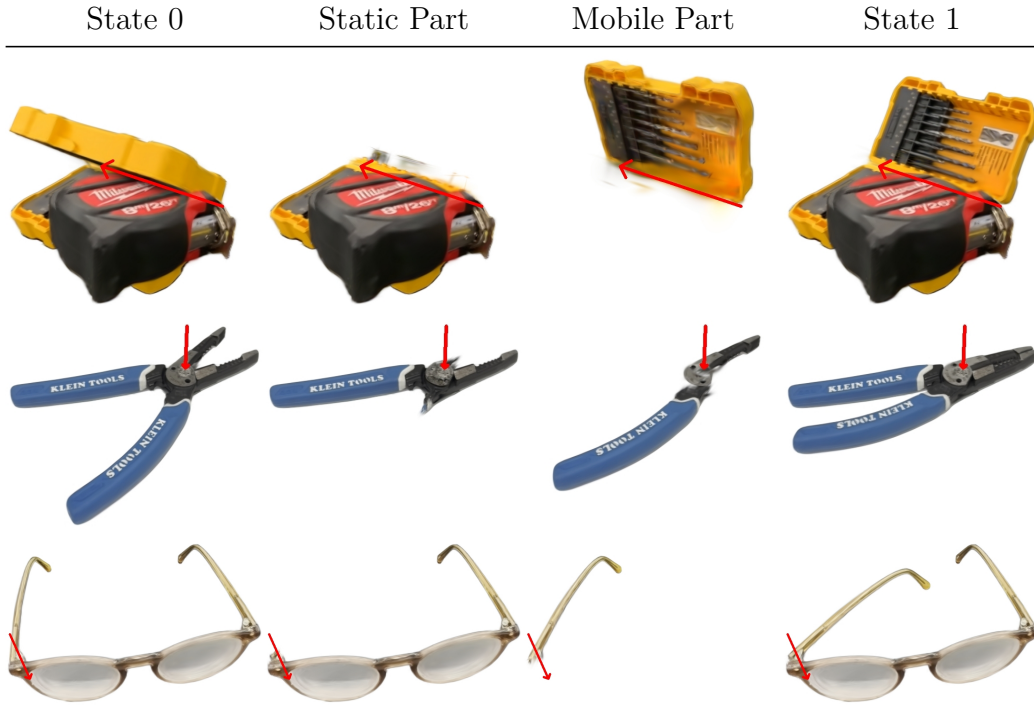


Figure 3.9: Real-world articulated object reconstructions.

parameters, potentially exacerbating the local minimum issue. Critically, existing methods that address multi-part scenarios typically require non-hierarchical articulations, where all mobile parts are directly connected to a single static parent part. Furthermore, they often rely on ground-truth depth and/or pre-trained priors [157], limiting their practicality. Consequently, our current work focuses on two-part articulated objects, whose real-world prevalence warrants a dedicated solution. Nevertheless, adapting SPLART for multi-part articulations remains a promising direction for future research.

### 3.6.3 Relation to Subsequent Work

Because SPLART shares a narrow publication window with several closely-related methods, much of what we call “subsequent” is in fact concurrent. ArtGS [172], REArtGS [178], ArticulatedGS [169], ArtiPoint [177], Articulate-Anything [170], and Real2Code [173] all appeared around the same time as SPLART and differ primarily in which axis of capability they priori-

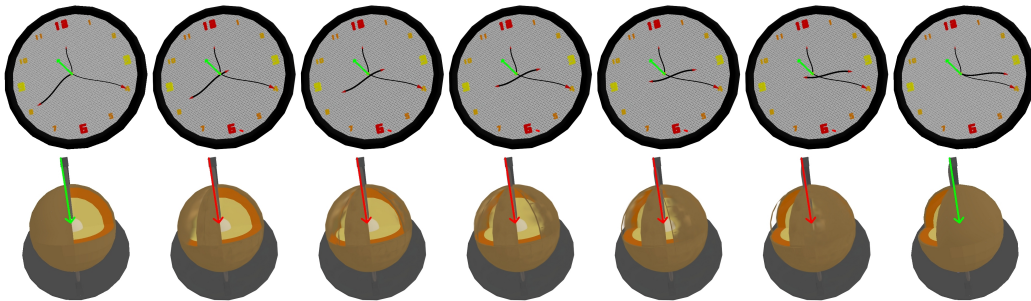


Figure 3.10: Failure cases due to inherent ambiguities in articulation.

size: multi-part topologies (ArtGS), VLM- or category-driven semantic priors (Articulate-Anything, Real2Code), or dynamic in-the-wild capture via point tracking (ArtiPoint). Methods that exceed SPLART on richer topologies or category coverage typically do so by accepting stronger supervision: pretrained semantic backbones, category priors, or dense temporal observations. Methods that further reduce data requirements, conversely, typically trade generality for category specialization. SPLART remains the reference design for its particular regime: joint part-level reconstruction and articulation inference from exactly two end-state observations, without category priors, 3D supervision, or pretrained semantic backbones. The natural continuation, extending this regime to multi-part topologies without sacrificing self-supervision, is the future direction sketched above.

### 3.7 Conclusion

This chapter introduced SPLART, a self-supervised framework that addresses the critical challenge of understanding the dynamic, functional properties of objects essential for robotic interaction. We have demonstrated that by leveraging 3D Gaussian Splatting and observing an object in just two distinct articulation states, it is possible to reconstruct its part-level geometry and appearance while simultaneously inferring its underlying kinematic model. The key contributions of this work—the augmentation of 3D Gaussians with a differentiable mobility parameter, a robust multi-stage optimization strategy, and the use of geometric self-supervision—collectively enable SPLART to achieve state-of-the-art performance without relying on 3D ground truth or category-specific priors.

The capabilities developed in this chapter represent a significant step forward in building an actionable world model for an embodied agent. While the previous chapter’s work on NeRFuser provided the foundational ability to construct a spatially-scalable, static map of the environment, SPLART enriches this representation with a crucial layer of temporal and functional understanding. By moving beyond static geometry to model the articulated nature of objects, we equip the robot with the knowledge not just of *what* is in the world, but *how* it can be manipulated.

With the spatial and kinematic layers of our world model now in place, the final piece of the puzzle is to connect them to high-level embodied intelligence. Having established *how* to build a large-scale map and *how* to understand the movable objects it contains, the next chapter will bridge the gap to robotic action by leveraging the reasoning capabilities of Large Language Models. We will explore how an LLM can serve as a central reasoning engine to interpret complex human commands and maintain a consistent world state over time, thereby translating high-level goals into concrete, physically-grounded actions that an embodied agent can execute.

## Chapter 4

# Bridging Perception and Action with LLM-Powered Embodied Reasoning

### 4.1 Introduction

The preceding chapters established a comprehensive perceptual foundation for an autonomous agent. In Chapter 2, we introduced NeRFuser, a framework to construct spatially-scalable, high-fidelity 3D models of static environments. Subsequently, in Chapter 3, we enriched this world model with a crucial layer of temporal and functional understanding, enabling the reconstruction of articulated objects and the inference of their kinematic properties. Together, these contributions provide a robot with a rich, detailed, and dynamic representation of its surroundings, one where it can perceive not only *what* exists in the world and *where*, but also *how* objects can move and be manipulated.

However, this powerful perceptual apparatus is only one half of the equation for true embodied intelligence. With the spatial and kinematic layers of our world model in place, the final and most critical challenge is to bridge the gap between perception and action. How can a robot leverage this detailed understanding to reason about its environment, comprehend high-level human instructions, and execute complex, multi-step tasks that unfold over extended interactions? This chapter posits that the emergent reasoning capabilities of Large Language Models (LLMs) provide the key to unlocking

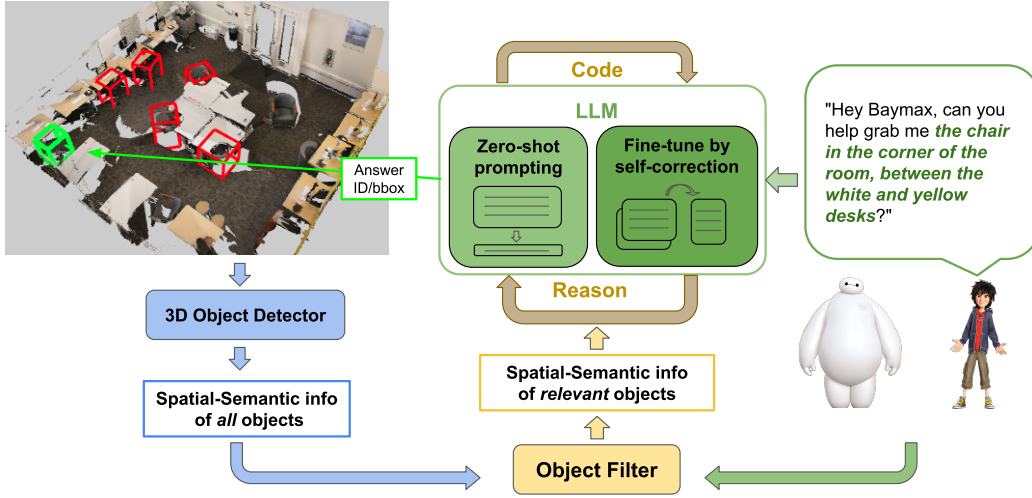


Figure 4.1: The overall **Transcrib3D** framework, which takes as input the colored point-cloud and referring expression (in teal), and outputs the ID or bounding box of the referent object. To resolve the referring expression “*the chair in the corner of the room, between the white and yellow desks*”, the framework needs to locate the pillow in the green box, while all other pillows in red boxes are distractors.

this connection, serving as a central cognitive engine that translates human goals into physically-grounded robotic behavior.

To this end, we address two fundamental and complementary challenges at the heart of embodied reasoning. First, a robot must be able to robustly interpret human language. Natural language is often ambiguous and context-dependent, featuring complex referring expressions (e.g., “the cup farthest from the door”) that require sophisticated spatial and semantic understanding to ground correctly in a cluttered 3D scene. Second, a robot must be able to perform long-horizon tasks that require maintaining a consistent understanding of the world as it changes over time, often as a direct result of the robot’s own actions. Without an explicit mechanism to track this evolving world state, reasoning can quickly become untethered from reality, leading to task failure.

This chapter presents a two-pronged approach that leverages LLMs to solve these intertwined problems. We introduce two distinct yet synergistic systems that, together, form a powerful framework for language-driven embodied reasoning:

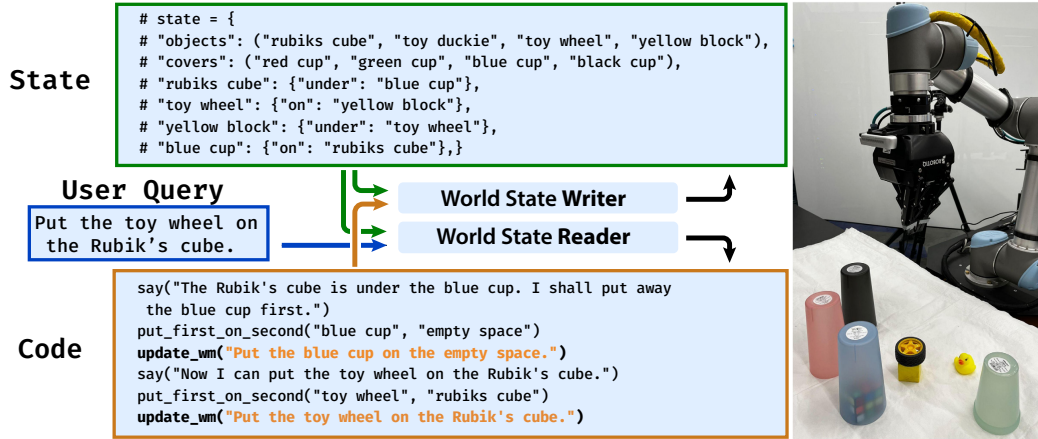


Figure 4.2: Our Statler framework enables robots to carry out complex tasks specified in natural language that require reasoning over long time horizons. Integral to our model are its world-state writer and world-state reader, two instances of general LLMs responsible for maintaining the explicit world state and generating code that enables the robot to carry out the task.

- **Transcrib3D**,<sup>1</sup> a system that tackles the challenge of language grounding by using text as a unifying medium [138] (see Figure 4.1). Rather than attempting to learn a complex, multi-modal mapping from vision and language to action—a process hampered by the scarcity of annotated 3D data—Transcrib3D first uses a 3D object detector to generate a textual “transcript” of the scene. This transcript, which details the properties and spatial relationships of detected objects, is then fed to an LLM, which can apply its powerful linguistic reasoning abilities to resolve complex, free-form referring expressions and accurately identify the user’s intended object of interest.
- **Statler**,<sup>2</sup> a novel architecture that endows an LLM-based agent with the ability to explicitly maintain and reason over the world state for long-horizon tasks [162] (see Figure 4.2). Addressing the limitations of prior “model-free” approaches that condition actions only on history, Statler employs a *world-state reader* to generate actions based on the current state and a *world-state writer* to update that state after an

<sup>1</sup>Code is publicly available at <https://github.com/ripl/Transcrib3D>.

<sup>2</sup>Code is publicly available at <https://github.com/ripl/statler>.

action is executed. By making the world state an explicit, persistent component of the reasoning process, Statler enables a robot to perform complex, sequential tasks that depend on a consistent and accurate understanding of the environment over time.

Transcrib3D and Statler address complementary challenges that together span the path from high-level human command to successful robotic execution. Transcrib3D targets the interpretive front-end, robustly grounding the user’s intent within the rich 3D world model. Statler targets the stateful execution back-end, enabling the robot to carry out multi-step plans by reasoning over a consistently updated world state. We develop and evaluate the two systems independently rather than as a single jointly deployed pipeline; we return to the question of how they could be integrated into a unified system in Section 4.5.3. By presenting these two systems, this chapter demonstrates how the abstract reasoning power of LLMs can be effectively harnessed to create robotic agents that are not only perceptually aware but also capable of intelligent, goal-directed action in complex, human-centric environments.

## 4.2 Related Work

The endeavor to imbue robotic agents with high-level reasoning by leveraging Large Language Models (LLMs) is a rapidly advancing frontier of research. Our work, which focuses on grounding natural language and maintaining world state, is situated at the confluence of several key areas: the application of LLMs to robotics, the specific challenges of language grounding in 3D environments, and the critical role of state representation in long-horizon planning and reasoning.

### 4.2.1 Large Language Models for Robotic Reasoning and Planning

Since LLMs are trained on internet-scale corpora, their infused common sense knowledge has proven invaluable for high-level planning from natural language instructions [52, 61, 105]. This has led to successful applications in both complex object manipulation [114, 122] and navigation tasks [29, 92, 101, 117]. A significant thrust in this area involves combining LLMs with



expressive visual-language embeddings to create powerful, general multi-modal embodied models [44, 83, 89]. More recently, this line has culminated in end-to-end vision-language-action (VLA) models that directly predict robot actions from language and visual inputs [87, 131, 140, 147, 152]. In parallel, the underlying LLM reasoning capabilities have themselves advanced rapidly, with structured frameworks such as Tree of Thoughts [125] and Graph of Thoughts [130] extending chain-of-thought prompting, and reasoning-specialized models trained via reinforcement learning [167] producing further gains on complex multi-step problems.

A particularly effective paradigm for translating an LLM’s reasoning into physical action is through code generation. Code generation is one of the most successful applications of LLMs [25, 34, 36, 55, 68, 112], and because code can interface with executable APIs for computation, vision, and manipulation, many works have focused on using LLMs as “tool-users” [113, 115, 118]. In the robotics context, Code-as-Policies [105] was among the first to demonstrate that an LLM can generate Python code that calls upon a library of robotic skills to solve complex tasks. Beyond action code, subsequent work has extended this paradigm to the specification of reward functions and affordance-like value maps for robot control [102, 110, 127, 158]; for a broader overview of this rapidly-evolving area, we refer the reader to recent surveys [139, 144]. Our work builds upon this powerful paradigm, using code generation as a primary mechanism for both querying the world state and for executing actions.

### 4.2.2 Grounding Language in 3D Scenes

For an LLM’s reasoning to be useful, it must be grounded in the robot’s physical environment. A primary challenge in this domain is the resolution of 3D referring expressions, a task popularized by benchmarks like ReferIt3D [24] and ScanRefer [26]. The dominant approach has been to develop end-to-end models that learn a joint multi-modal embedding, aggregating features from point-clouds, images, and text using Transformer architectures [11]. Methods such as MVT [60], BUTD-DETR [64], SAT [49], HAM [57], ViL3DRel [58], and 3D-VisTA [129] fall into this category. Others have explored joint learning of captioning and grounding [54, 56] or neuro-symbolic frameworks [100]. However, these multi-modal approaches are often hampered by the relative scarcity of annotated 3D data compared to the 2D domain, which limits their reasoning capabilities.

In contrast, our work on Transcrib3D aligns with an alternative strategy that uses text as the unifying medium, thereby sidestepping the need to learn complex cross-modal representations from limited data. Instead of multi-modal fusion, our approach first transcribes the 3D scene into a detailed textual format, then applies the powerful, pre-existing reasoning capabilities of an LLM to this textual scene representation. This philosophy is shared by other works that leverage textual 3D information for tasks like indoor scene generation [91] and multi-robot navigation [126]. Concurrent work has also explored using LLMs to reason over textual scene data for grounding, albeit with different, more task-specific reasoning pipelines [160]. Furthermore, the iterative code generation and self-correction process in Transcrib3D is directly inspired by general LLM reasoning frameworks such as ReAct [82] and Reflexion [154].

While Transcrib3D relies on text, a parallel stream of research has sought to build unified “3D-LLMs” that directly ingest 3D data. Grounded 3D-LLM [134] introduces referent tokens to link language and 3D regions within a single model. To address the data scarcity issue, 3D-GRAND [159] provides a large-scale dataset for grounding and hallucination reduction. Similarly, LLM-Grounder [160] employs an LLM agent for open-vocabulary grounding. While these end-to-end approaches are promising, Transcrib3D demonstrates the efficacy of a text-bottleneck approach that leverages the vast reasoning priors of off-the-shelf LLMs without requiring expensive 3D pre-training.

### 4.2.3 State Maintenance for Long-Horizon Reasoning

While grounding a single command is crucial, enabling a robot to perform long-horizon tasks requires a persistent understanding of the world that evolves over time. However, recent work has shown that standard LLMs struggle with planning and maintaining consistency over long sequences of actions [107, 121, 155]. Many existing LLM-robotics systems are effectively “model-free,” conditioning each new action only on the history of observations and prior actions without an explicit, structured representation of the world state. This approach is not ideal, as the unstructured history can quickly grow to an unmanageable size.

Our work on Statler is motivated by the principle that an explicit state representation can significantly enhance reasoning and planning. This idea has been shown to be effective in algorithmic reasoning tasks, where spelling out intermediate steps or states improves final performance [41, 73]. In the

context of robotics, some approaches use LLMs as a high-level intermediary to interface with traditional, symbolic planners that operate on a formal state representation [97, 107, 123]. Others have used feedback or fine-tuning on world-model traces to improve planning performance [124, 155]. Statler proposes a different approach: empowering the LLM itself to maintain and update a natural language representation of the world state. By explicitly separating the roles of a *world-state reader* (which generates actions based on the current state) and a *world-state writer* (which updates the state after an action), Statler provides a structured yet flexible mechanism for tracking the dynamic environment, enabling more robust and reliable execution of multi-step, state-dependent tasks.

The critical role of explicit state tracking, central to Statler, is echoed in recent long-horizon planning literature. RoboHorizon [166] utilizes LLM-assisted “world models” to generate rewards for multi-stage manipulation. StateAct [153] introduces a “chain-of-states” approach, prompting the LLM to explicitly track state changes in a text-based history. Unlike these methods which often rely on implicit or unstructured state histories, Statler enforces a structured separation between state reading and writing, ensuring consistency over longer horizons.

## 4.3 Methodology

To bridge the gap between a rich 3D world model and intelligent robotic action, we introduce two complementary systems that leverage the reasoning power of Large Language Models (LLMs). The first, Transcrib3D, focuses on robustly grounding a user’s natural language commands in the 3D scene. The second, Statler, addresses execution and planning by maintaining an explicit world state for long-horizon, multi-step tasks. This section details the technical approach of each system, which together form a comprehensive framework for LLM-powered embodied reasoning.

### 4.3.1 Transcrib3D: Grounding Referring Expressions via Scene Transcription

The Transcrib3D framework is designed to resolve complex 3D referring expressions by using text as a unifying medium, effectively bridging 3D perception with LLM-based reasoning. As illustrated in Figure 4.1, the overall

pipeline takes a colored point-cloud as input. It first applies a 3D object detector to generate an exhaustive list of objects in the scene, which is then transcribed into a textual format. This list is subsequently filtered to identify objects relevant to the provided referring expression. The resulting filtered object information, along with the expression itself, serves as input to an LLM-based reasoning module that iteratively generates and executes code to arrive at the final answer. We consider two primary modes for interfacing with the LLM: a zero-shot approach guided by carefully designed principles, and a more adaptive approach involving the fine-tuning of smaller models based on self-reasoned corrections.

### Detect and Transcribe 3D Information

Given a colored point-cloud of the scene, we first perform semantic segmentation of objects using Mask3D [116]. We associate with each detected object its category based on the semantics, center location, spatial extent according to the 3D bounding box, as well as its mean color. The 3D orientation of an object can also be incorporated using PartNet [17]. Transcrib3D compiles the information associated with all detected objects as a list to form an object-centric scene description. The following shows an example of such a scene transcript (exact numeric values and duplicate object categories are replaced by ...).

#### Spatial Semantic Scene Description for All Objects

```
scene0592: ... Scene center: [...] ... objs list:
monitor, id=0, ctr=[...], size=[...], rgb=[...];
...
box, id=5, ctr=[...], size=[...], rgb=[...];
copier, id=6, ctr=[...], size=[...], rgb=[...];
floor, id=7, ctr=[...], size=[...], rgb=[...];
wall, id=8, ctr=[...], size=[...], rgb=[...];
...
armchair, id=15, ctr=[...], size=[...], rgb=[...];
```

### Pre-Filtering Relevant Objects for Utterance

The aforementioned procedure results in a representation of every object detected in the scene. However, only a small fraction of these objects will typically be relevant to the given referring expression. Given the utterance,

“chair in the corner of the room, between white and yello desks”, the model identifies the following objects as relevant,

**Spatial Semantic Scene Description for Relevant Objects**

```
scene0592: ... Scene center: [...] ... objs list:
wall, id=8, ctr=[...], size=[...], rgb=[...];
...
armchair, id=15, ctr=[...], size=[...], rgb=[...];
...
chair, id=19, ctr=[...], size=[...], rgb=[...];
```

and the remainder as irrelevant. Simplifying the object list by filtering out irrelevant items (taking into account synonyms and hypernyms) not only reduces processing time and the number of tokens for LLMs, but also facilitates reasoning by reducing content that may potentially be distracting. Such an approach has been shown to improve the efficiency of language grounding [19].

### Iterative Code Generation and Reasoning

Compositional reasoning that involves arithmetic calculations, which is crucial for spatial reasoning, is a well-known weakness of Transformers and LLMs [90]. However, most contemporary approaches to 3D referring expression resolution rely on a single forward pass of a Transformer to reach the final answer, which can be limiting in terms of reasoning power. In order to avoid this weakness, Transcrib3D equips the LLM with a Python interpreter and directs the LLM to generate code whenever quantitative evaluations are necessary. Transcrib3D then locally executes the generated code using a Python interpreter and appends the output to the original conversation. The resulting prompt is sent back to the LLM to generate the next response. If there is any error in executing the generated code, Transcrib3D also feeds information about the error back to the LLM for re-generation. This process continues until the LLM believes that the reasoning is complete. Figure 4.3 provides an illustration of this process.

We want to emphasize that our design does not distinguish between code generation and chat reasoning steps. Such a framework simplifies the design, allows for flexible reasoning, and makes it easy for fine-tuning. But in turn, it requires the LLM to have the ability to do both, so language models that are specifically trained for only coding or only chatting will not perform the best within our framework.

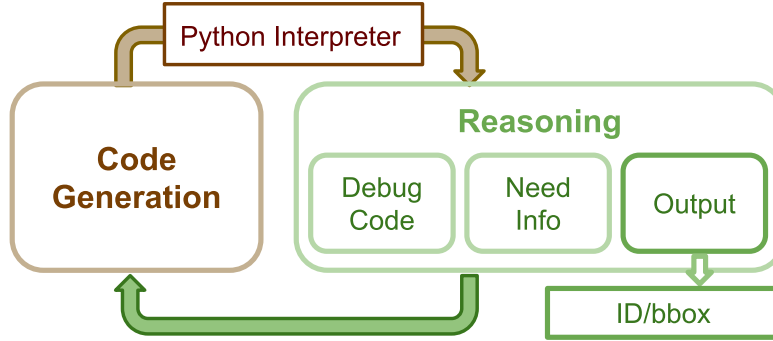


Figure 4.3: Illustration of the iterative code generation and reasoning process in Transcrib3D. After code generation, execution results from a local Python interpreter are fed back to the LLM for further reasoning. The LLM then proceeds to either (1) fix code errors when encountered, (2) generate additional code to obtain more information, or (3) output the referred objects if it has all the needed information. This process continues until the LLM believes the reasoning to be complete.

### Principles-Guided Zero-Shot Prompting

Spatial reasoning can require special geometric operations with respect to natural languages that are not familiar with LLMs [90]. For example, when handling left/right spatial relations, which should factor in the observer’s viewpoint, LLMs often naively ground them with smaller/larger values in the  $x$ -coordinates. In order to overcome these deficiencies, Transcrib3D employs a suite of general principles to guide LLM reasoning. For example: (1) use HSL space rather than RGB space to better match colors with its names; (2) perform vector operations to resolve directional relations; and (3) calculate the point-to-plane distance to determine who’s closer to a wall. The full prompts are included in the released code. Although the set of principles work generally well once designed, new environments and new dataset will require certain new set of rules to be adaptive, which makes it labor-intensive and hard to scalable.

### Fine-tuning from Self-Reasoned Correction

To overcome the shortcomings of a human-designed rule-based system and to make the method more adaptive to new environments, we leverage the self-

introspection capabilities of LLMs to reflect on their mistakes. By collecting these self-reflections and fine-tuning the model based on the insights gained, the system continuously learns, improves, and adapts to new challenges. We do so via the following procedure:

1. Use the prompt with general principles on the training set to have the LLM generate an initial set of answers, each accompanied by an elaborated reasoning process.
2. For any incorrect answer, augment the original prompt with the correct object ID and request the LLM to reflect on why the original answer is incorrect (“What went wrong?”).
3. Append the reflected reasoning, without the answer, to the original prompt to run the reasoning module again.
4. Gather the reasoning processes of the correct examples from the original process and the self-reflected examples to produce a dataset for LLM fine-tuning.
5. Use supervised fine-tuning with quantization [70] to fine-tune on small-scale language models [120].

Listing 4.1 shows an example of the self-reasoned correction step. Note that after fine-tuning, we no longer include general principles in the prompt. In this way, the system not only incorporates the guiding principles but further improves itself by learning from self-reasoned correction.

### Application to Robotic Manipulation

The ultimate goal of grounding language is to enable embodied agents to follow human instructions. The Transcrib3D framework can be directly integrated into a robotic manipulation pipeline to achieve this. A language-guided pick-and-place task, for instance, can be decomposed into three steps: (1) parsing the instruction to identify the referring expressions for the object to be picked and the target placement location; (2) resolving these two expressions in the context of the robot’s 3D environment using the Transcrib3D methodology; and (3) executing the physical pick-and-place action. As demonstrated in Figure 4.4, this integration allows a real robot to successfully interpret and execute commands involving sophisticated spatial and relational language, showcasing the practical utility of our approach.

**Self-Correction for Fine-Tuning**

*Assistant:* Based on the calculated metrics, we can see that chair 49 has the smallest distance to the nearest two walls, which means it is most likely to be in the corner of the room. It also has the smallest distance to the nearest white desk and a relatively small distance to the nearest yellow desk, which means it is likely to be between a white desk and a yellow desk. Therefore, chair 49 is the most likely to be the chair described in the description. Now the answer is complete – ‘ID’:49

*User:* The correct answer is object 18. Can you double check the information of object 18 and the given prompt and see where you got wrong?

*Assistant:* Let’s revisit the information provided for object 18 and the constraints given in the prompt to identify where the mistake might have occurred. Object 18 is described as follows: - Type: Chair - Center: [-2.98, -3.31, 0.39] - Size: [0.53, 0.61, 0.81] - RGB: [60, 58, 50] The constraints from the prompt are: 1. The chair is in the corner of the room. 2. The chair is between a white and a yellow desk. Upon reviewing the code execution results, it was noted that chair 18 has a relatively small distance to the nearest two walls, indicating it could be in the corner of the room. Additionally, chair 18 has a very small distance to the nearest white desk (0.66) and a reasonable distance to the nearest yellow desk (2.58), which aligns with the constraint of being between a white and a yellow desk. **The mistake in the initial analysis was not properly considering the combination of both constraints together.** While chair 49 had the smallest distance to the nearest two walls and a very small distance to the nearest white desk, it did not optimally satisfy the condition of being between a white and a yellow desk as well as chair 18 did. Chair 18’s proximity to both a white and a yellow desk, along with its corner location, makes it the correct answer according to the given constraints.

Listing 4.1: An example of self-reasoned correction for fine-tuning. During self-correction, the LLM realizes its mistake during the initial analysis.

### 4.3.2 Statler: State-Maintaining Models for Embodied Reasoning

While Transcrib3D grounds language to identify objects, Statler focuses on using an LLM to execute complex, multi-step tasks that require maintaining a consistent understanding of the world as it evolves over time. The key idea is to explicitly maintain an estimate of the world state and to track its transitions as actions are taken by the agent. This model-based approach addresses the known challenge that LLMs often struggle to implicitly maintain



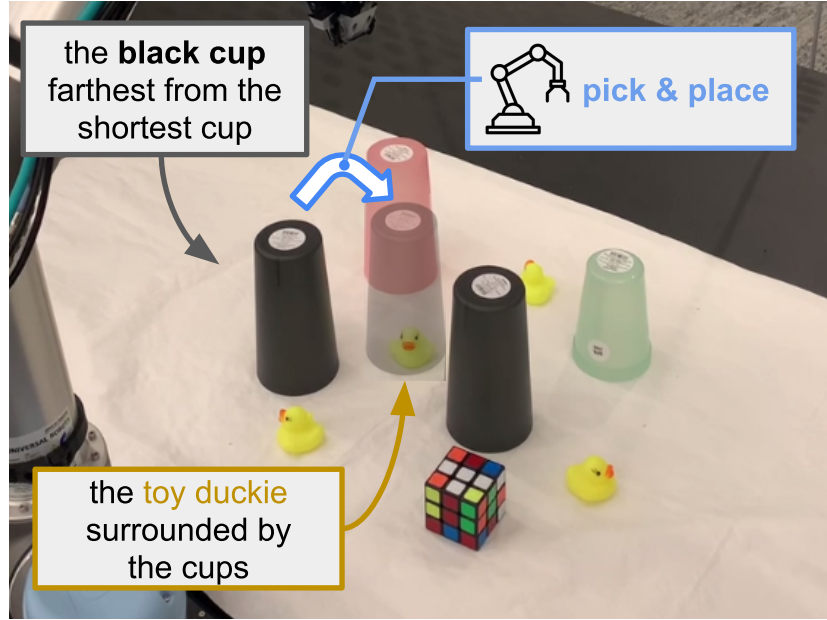


Figure 4.4: Application of Transcrib3D to a real-world robotic manipulation task. The system enables a robot to successfully execute a pick-and-place instruction containing complex referring expressions: “cover *the toy duckie surrounded by the cups* with the black cup farthest from the shortest cup.”

state over long interaction horizons [85, 149].

### Motivation: The Need for Explicit State Tracking

Recent work has demonstrated that LLMs struggle with planning and maintaining consistency over long sequences of actions [107, 121, 155]. When an LLM must implicitly maintain the world state in its internal representation, its performance can degrade rapidly as the number of steps increases.

To illustrate this limitation, we use the simple shell game of *three-cups-and-a-ball* as a pedagogical example. In this game, a ball is hidden under one of three cups. After a series of  $K$  swaps between pairs of cups, the model must identify the final location of the ball. We compare three prompting strategies: a vanilla approach that asks for the final state directly (Listing 4.2); a Chain-of-Thought (CoT) approach that asks the model to reason through all intermediate states at the end (Listing 4.3); and our proposed state-maintaining approach, where the model is prompted to update the state

Listing 4.2: The prompt and `desired output` of the vanilla LLM.

---

```

1 # Initial state
2 cups = [False, True, False]
3 Swapping cup 1 with cup 2
4 Swapping cup 0 with cup 2
5 Swapping cup 1 with cup 2
6 cups = [True, False, False]

```

---

Listing 4.3: The prompt and `desired output` of LLM+CoT.

---

```

1 # Initial state
2 cups = [False, True, False]
3 Swapping cup 1 with cup 2
4 Swapping cup 0 with cup 2
5 Swapping cup 1 with cup 2
6 cups = [False, False, True]
7 cups = [True, False, False]
8 cups = [True, False, False]

```

---

Listing 4.4: The prompt and `desired output` of LLM+State.

---

```

1 # Initial state
2 cups = [False, True, False]
3 Swapping cup 1 with cup 2
4 cups = [False, False, True]
5 Swapping cup 0 with cup 2
6 cups = [True, False, False]
7 Swapping cup 1 with cup 2
8 cups = [True, False, False]

```

---

after each individual swap (Listing 4.4).

As shown in Figure 4.5, the state-maintaining method significantly outperforms the other approaches. The accuracy of the vanilla and CoT methods deteriorates quickly as the number of swaps increases, whereas the state-maintaining method remains robust. This simple experiment clearly demonstrates that by making the world state an explicit, external artifact that is updated at each step, we can drastically improve an LLM’s ability to perform long-horizon, state-dependent reasoning. This core principle motivates the design of the Statler framework.

### The Statler Framework: A Reader-Writer Architecture

The Statler framework, illustrated in Figure 4.2, operationalizes the principle of explicit state maintenance by splitting the total reasoning burden across two distinct, prompted LLMs: a **world-state reader** and a **world-state writer**. This separation of concerns allows each component to become specialized for its respective subtask. The world-state reader is responsible for responding to the user’s query by generating an action (in the form of executable code) conditioned on the current world state. The world-state writer is responsible for updating the state representation after an action has been executed. This approach can be seen as a model-based extension of the Code-as-Policies (Code-as-Policies) paradigm [105], incorporating an explicit state-tracking mechanism while retaining the power of hierarchical code generation.

To understand the operation of the reader and writer models, consider the example prompts in Listings 4.5 and 4.6. In Listing 4.5, the world-state reader is given an initial state representation (in a JSON-like format) and

Listing 4.5: World-state reader. The text highlighted in green represents the part that the model is expected to generate.

---

```

1 # state = {
2 #   "objects": ["cyan block", "yellow block", "brown block", "purple block", "blue
   block", "green bowl", "red bowl", "disinfector"],
3 #   "relations": [],
4 #   "disinfector": {"contains": []},
5 #   "cyan block": {"is": ["dirty"]},
6 #   "yellow block": {"is": ["clean"]},
7 #   "brown block": {"is": ["clean"]},
8 #   "purple block": {"is": ["dirty"]},
9 #   "blue block": {"is": ["clean"]},
10 #   "green bowl": {},
11 #   "red bowl": {}
12 # }
13 # query: Put the cyan block on the yellow block
14 put_first_on_second("cyan block", "yellow block")
15 update_wm("Put the cyan block on the yellow block")

```

---

Listing 4.6: World-state writer. The text rendered in blue highlights the updated part of the state.

---

```

1 # state = {
2 #   "objects": ["cyan block", "yellow block", "brown block", "purple block", "blue
   block", "green bowl", "red bowl", "disinfector"],
3 #   "relations": [],
4 #   "disinfector": {"contains": []},
5 #   "cyan block": {"is": ["dirty"]},
6 #   "yellow block": {"is": ["clean"]},
7 #   "brown block": {"is": ["clean"]},
8 #   "purple block": {"is": ["dirty"]},
9 #   "blue block": {"is": ["clean"]},
10 #   "green bowl": {},
11 #   "red bowl": {}
12 # }
13 # query: Put the cyan block on the yellow block.
14 # state = {
15 #   "objects": ["cyan block", "yellow block", "brown block", "purple block", "blue
   block", "green bowl", "red bowl", "disinfector"],
16 #   "relations": [{"cyan block is on yellow block"}],
17 #   "disinfector": {"contains": []},
18 #   "cyan block": {"is": ["dirty"]},
19 #   "yellow block": {"is": ["dirty"]},
20 #   "brown block": {"is": ["clean"]},
21 #   "purple block": {"is": ["dirty"]},
22 #   "blue block": {"is": ["clean"]},
23 #   "green bowl": {},
24 #   "red bowl": {},
25 # }

```

---

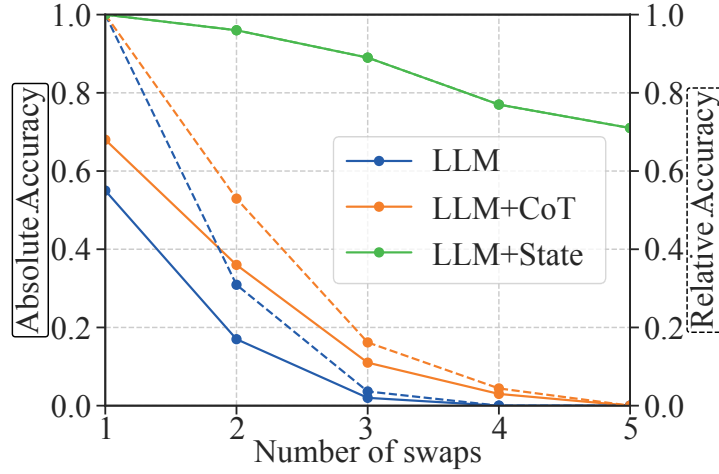


Figure 4.5: Model accuracies on the *three-cups-and-a-ball* shell game. The state-maintaining method significantly outperforms both vanilla and Chain-of-Thought (CoT) prompting, with the performance gap increasing as the number of swaps (the task horizon) grows.

a user query (“Put the cyan block on the yellow block”). The reader’s task is to generate the Python code that accomplishes this command, taking the current state into account. After this code is generated and executed, the state may need to be updated. The generated code can trigger this update by calling a special `update_wm` function, which passes a new query to the world-state writer.

Listing 4.6 shows the corresponding input for the world-state writer. It receives the current state and the update query from the reader’s generated code. Its sole responsibility is to generate the new, updated JSON-like representation of the world state. This updated state is then saved and becomes the “current state” for the next turn of interaction. By explicitly passing the state back and forth between the reader and writer, Statler ensures that all actions are grounded in a consistent and up-to-date world state, enabling more robust performance on long-horizon, state-dependent tasks.

## 4.4 Experiments

To validate the effectiveness of our LLM-powered framework for embodied reasoning, we conducted a comprehensive set of experiments designed to evaluate both the language grounding capabilities of Transcrib3D and the long-horizon, state-maintaining performance of Statler. This section details the experimental setups, datasets, and results for each system, demonstrating their individual strengths and collective contribution to enabling sophisticated robotic intelligence.

### 4.4.1 Evaluating Transcrib3D: 3D Referring Expression Resolution

We first assess the performance of Transcrib3D on the task of grounding complex referring expressions in 3D scenes. Our evaluation uses standard benchmarks to quantify its accuracy and compares it against state-of-the-art multi-modal approaches.

#### Benchmarks and Metrics

We evaluate Transcrib3D on two widely-used benchmarks:

- **ReferIt3D** [24]: This benchmark formulates the task as a multiple-choice problem where, given a set of segmented objects, the goal is to identify the unique referent from a natural language expression. It includes two distinct subsets: SR3D, with template-based utterances, and NR3D, with free-form human-generated utterances. Performance is measured by accuracy.
- **ScanRefer** [26]: In this benchmark, the model is given a colored point-cloud and a query and must return the 3D bounding box of the target object. Performance is measured by accuracy at different Intersection-over-Union (IoU) thresholds with the ground-truth bounding box, specifically at 25% (acc@0.25IoU) and 50% (acc@0.5IoU).

#### Grounding Accuracy on ReferIt3D

We tested several variations of our method against baselines on the ReferIt3D dataset. On NR3D, our best model, Transcrib3D (GPT4-P), was evaluated

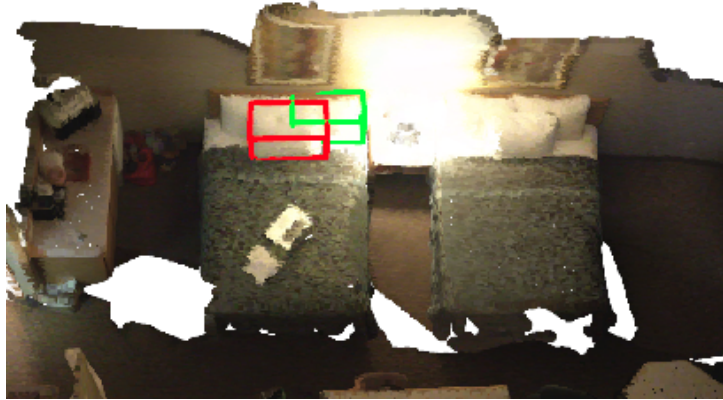
| Method                      | Nr3D        |             |             |             |             | Sr3D        |             |             |             |             |
|-----------------------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
|                             | Overall     | Easy        | Hard        | View Dep.   | View Ind.   | Overall     | Easy        | Hard        | View Dep.   | View Ind.   |
| SAT <sup>†</sup> [49]       | 49.2        | 56.3        | 42.4        | 46.9        | 50.4        | 57.9        | 61.2        | 50.0        | 49.2        | 58.3        |
| BUTD-DETR <sup>†</sup> [64] | 54.6        | 60.7        | 48.4        | 46.0        | 78.0        | 67.0        | 68.6        | 63.2        | 57.0        | 67.7        |
| MVT <sup>†</sup> [60]       | 59.5        | 67.4        | 52.7        | 59.1        | 60.3        | 64.5        | 66.9        | 58.8        | 58.4        | 64.7        |
| ViL3DRel <sup>§</sup> [58]  | 64.4        | 70.2        | 57.4        | <b>62.0</b> | 64.5        | 72.8        | 74.9        | 67.9        | 63.8        | 73.2        |
| 3D-VisTA <sup>§</sup> [129] | 64.2        | 72.1        | 56.7        | 61.5        | 65.1        | 76.4        | 78.8        | 71.3        | 58.9        | 77.3        |
| Transcrib3D (GPT-3.5-NP)    | 33.8        | 42.6        | 25.0        | 26.9        | 36.0        | 79.3        | 82.8        | 70.7        | 70.6        | 80.5        |
| Transcrib3D (GPT-3.5-P)     | 46.6        | 56.0        | 37.1        | 29.9        | 51.9        | 80.0        | 80.8        | 78.1        | 82.4        | 79.7        |
| Transcrib3D (GPT-4-NP)      | 64.5        | 71.8        | 57.1        | 49.4        | 71.3        | 97.4        | 98.4        | 94.7        | 87.3        | <b>98.7</b> |
| Transcrib3D (GPT-4-P)       | <b>70.2</b> | <b>79.7</b> | <b>60.3</b> | 60.1        | <b>75.4</b> | <b>98.4</b> | <b>99.2</b> | <b>96.2</b> | <b>98.2</b> | 98.4        |

Table 4.1: Grounding accuracy (%) on Nr3D and Sr3D. <sup>†</sup> denotes results from the official benchmarks, and <sup>§</sup> denotes results reported in the respective papers. **P**: with principles; **NP**: no principles. **Easy** samples have one or no distractors in a scene, while **Hard** samples have two or more. **View Dep.** (view-dependent) samples contain language that relies on viewing directions, while **View Ind.** (view-independent) samples do not. All our models are equipped with interactive code generation. Transcrib3D with GPT-4 and general principles surpasses all baselines by a large margin.

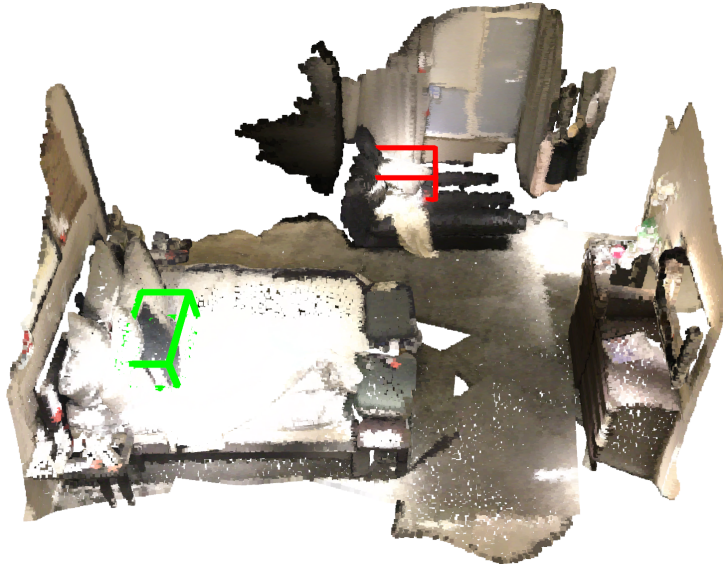
on the full test set, while other variants were evaluated on a subset of 500 random samples. On SR3D, all Transcrib3D models were evaluated on a similar 500-sample subset. Table 4.1 presents the results. Our approach significantly outperforms previous methods, showcasing the power of leveraging LLM reasoning over a textual scene transcript. Figure 4.6 provides a qualitative comparison, highlighting Transcrib3D’s superior ability to handle complex, multi-constraint expressions.

### Grounding Accuracy on ScanRefer

We evaluated our best variant (GPT4-P) on the ScanRefer benchmark, which requires predicting a bounding box without ground-truth segmentations. Table 4.2 shows that Transcrib3D achieves state-of-the-art performance, whether using detected or ground-truth bounding boxes as input. The performance gain is particularly notable when using ground-truth boxes, suggesting that improvements in 3D object detection will directly translate to better grounding performance for our system.



(a) Utterance: “On the bed with the laptop its the pillow by the light.” This query requires understanding three constraints together: “On the bed”, “with the laptop”, and “by the light”.



(b) Utterance: “The foremost pillow on the bed of the group of pillows.” This query requires understanding that the pillow needs to be among a group.

Figure 4.6: Qualitative comparisons between Transcrib3D (ours, in **green**) and 3D-VisTA (in **red**) on the NR3D dataset. Our method correctly grounds the complex language where the baseline fails.

| Method            | Data  | Det. | Overall  |         |
|-------------------|-------|------|----------|---------|
|                   |       |      | acc@0.25 | acc@0.5 |
| ViL3DRel [58]     | Full  | PG   | 47.9     | 37.7    |
| 3D-VisTA [129]    | Full  | M3D  | 50.6     | 45.8    |
| 3D-VisTA (re-run) | Full  | M3D  | 50.7     | 45.9    |
| 3D-VisTA (re-run) | Part. | M3D  | 50.6     | 44.6    |
| Transcrib3D       | Part. | M3D  | 51.2     | 44.4    |
| Transcrib3D + Cam | Part. | M3D  | 51.3     | 45.5    |
| 3D-VisTA (re-run) | Part. | GT   | 55.6     | 55.6    |
| Transcrib3D       | Part. | GT   | 62.0     | 62.0    |
| Transcrib3D + Cam | Part. | GT   | 64.2     | 64.2    |

Table 4.2: Grounding accuracy (%) on ScanRefer. **Full**: entire validation set (5,410 samples). **Part.**: fixed 500-sample subset common to all methods. **Det.**: 3D detector used. **PG** = PointGroup [27]; **M3D** = Mask3D [116]; **GT** = ground-truth boxes. On ScanNet, mAP@0.5 is 56.7 for PG and 73.7 for M3D [116]. We evaluate ours and re-run 3D-VisTA on the same 500 samples using M3D, GT boxes, and an additional “+ Cam” setting that augments the scene transcript with camera-view metadata from ScanRefer. ScanRefer permits all modalities and ranks methods together; our zero-shot method easily incorporates this extra information.

### Effects of Fine-tuning from Self-Reasoned Correction

To assess the viability of using smaller, more efficient models, we studied the effect of our proposed fine-tuning strategy. We used GPT-4 to generate self-corrected reasoning traces on the NR3D training set and then used this data to fine-tune a smaller `gpt-3.5-turbo` model. As shown in Table 4.3, the fine-tuned smaller model significantly closes the performance gap with the much larger GPT-4 model. This result is particularly promising for deploying Transcrib3D on robotic platforms with limited computational resources, as it demonstrates a clear path to achieving high performance without relying on large, proprietary models.



| Models                    | Total | Easy | Hard | Time  |
|---------------------------|-------|------|------|-------|
| GPT-3.5 (zero-shot)       | 33.8  | 42.6 | 24.0 | ~ 15s |
| GPT-3.5 (correct-only FT) | 60.7  | 52.8 | 41.9 | ~ 15s |
| GPT-3.5 (self-correct FT) | 61.5  | 54.6 | 80.0 | ~ 15s |
| GPT-4 (zero-shot)         | 69.4  | 78.7 | 60.0 | ~ 60s |

Table 4.3: Performance of fine-tuning models on NR3D. Fine-tuned GPT-3.5 models demonstrate a significant improvement in performance when compared to zero-shot models, closely approaching the capabilities of GPT-4. The model fine-tuned on self-corrected examples sees an increase in performance compared to that fine-tuned on only correct examples, particularly for hard queries. Notably, the fine-tuned models are not provided with rule-based prompts during both fine-tuning and inference time, suggesting that implicit decision rules are learned from examples.

#### 4.4.2 Evaluating Statler: Long-Horizon Embodied Reasoning

Having established a robust method for language grounding, we now turn to evaluating Statler’s ability to perform long-horizon, multi-step tasks that require maintaining an explicit world state.

##### Simulated Tabletop Manipulation Domains

We evaluate Statler on three simulated tabletop manipulation domains, each designed to test different aspects of state-dependent reasoning. The domains, illustrated in Figure 4.7, are:

- **Pick-and-Place:** Requires the robot to track object locations over a sequence of manipulation actions.
- **Block Disinfection:** Involves a latent property (cleanliness) that is not directly observable and must be inferred and tracked throughout the task as objects interact.
- **Relative Weight Reasoning:** Requires the agent to remember and reason about the relative weights of different objects in order to satisfy complex constraints.

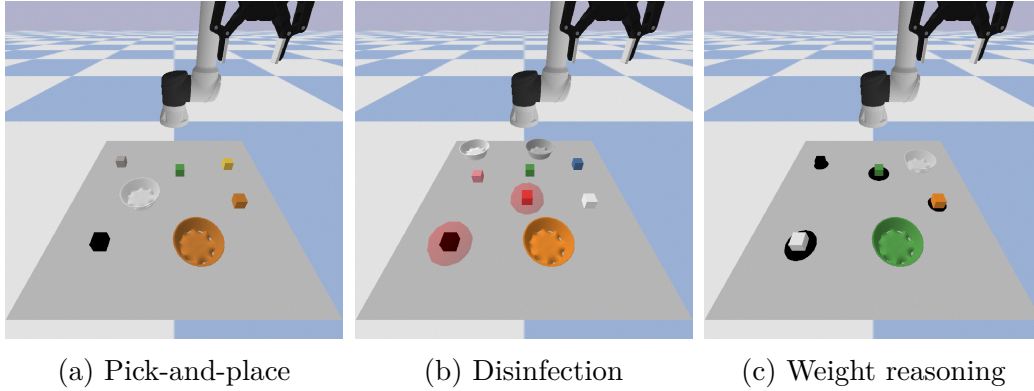


Figure 4.7: The simulated domains used to evaluate Statler. Latent properties, such as dirtiness in (b) (translucent sphere) and weight in (c) (disk radius), are not visible to the robot and must be tracked in the world state.

For each domain, we created 20 evaluation episodes, each consisting of 5-16 consecutive user queries that require reasoning over the interaction history. Appendix B presents one example episode per domain together with the prompts used.

### Comparison with Baselines

We compare Statler against two strong baselines: Code-as-Policies [105], a model-free approach that conditions actions on history, and CaP with Chain-of-Thought (CaP+CoT) [79], which encourages intermediate reasoning. An episode is considered successful only if the model correctly responds to all queries in the sequence.

Table 4.4 reports the episode and individual step success rates. The results are stark: while the baselines correctly handle more than half of the individual steps, their performance collapses over longer horizons, failing to complete any full episodes in the more complex domains. In contrast, Statler’s explicit state-maintaining approach allows it to complete a majority of episodes, demonstrating a dramatic improvement in long-horizon reasoning. Figures 4.8 and 4.9 shows qualitative examples where Code-as-Policies fails on state-dependent instructions that Statler handles correctly.

|                | Pick & Place       | Disinfection             | Weight             |
|----------------|--------------------|--------------------------|--------------------|
| CaP            | 0.00 (0.54)        | 0.00 (0.68)              | 0.00 (0.84)        |
| CaP+CoT        | 0.25 (0.76)        | 0.00 (0.20) <sup>†</sup> | 0.30 (0.88)        |
| Statler (ours) | <b>0.50 (0.88)</b> | <b>0.40 (0.82)</b>       | <b>0.55 (0.93)</b> |

Table 4.4: Episode and individual step (in parentheses) success rates for each sequential task. <sup>†</sup>marks cases where the context limit was often exceeded.

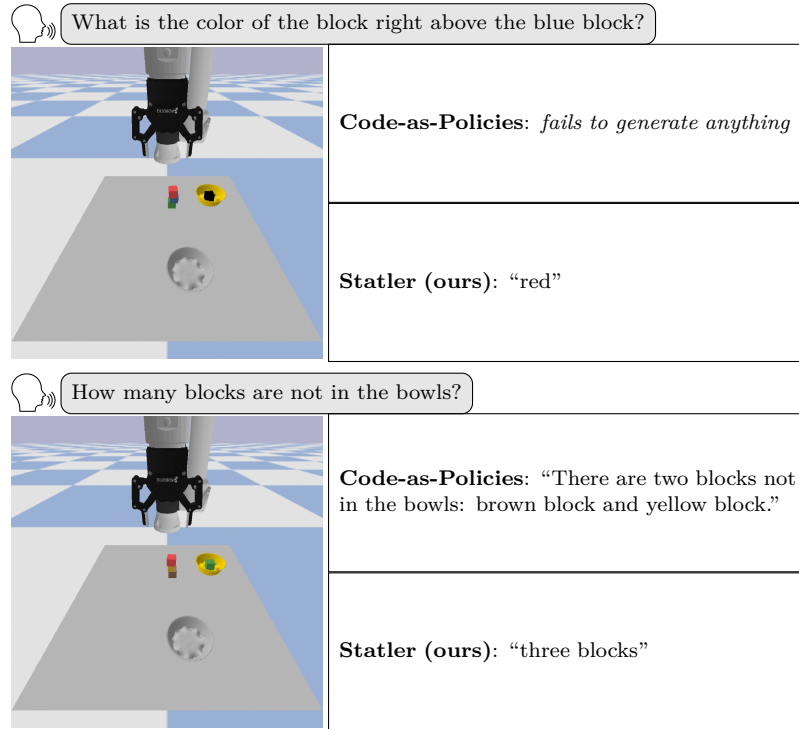


Figure 4.8: LLM queries with and without a maintained world state. In the first scenario, Code-as-Policies fails to produce an answer, whereas our Statler model returns the correct response. In the second, one block is occluded and Code-as-Policies incorrectly labels two blocks as “not a bowl.” By maintaining a persistent world state, our method remains aware of the third block and answers correctly.

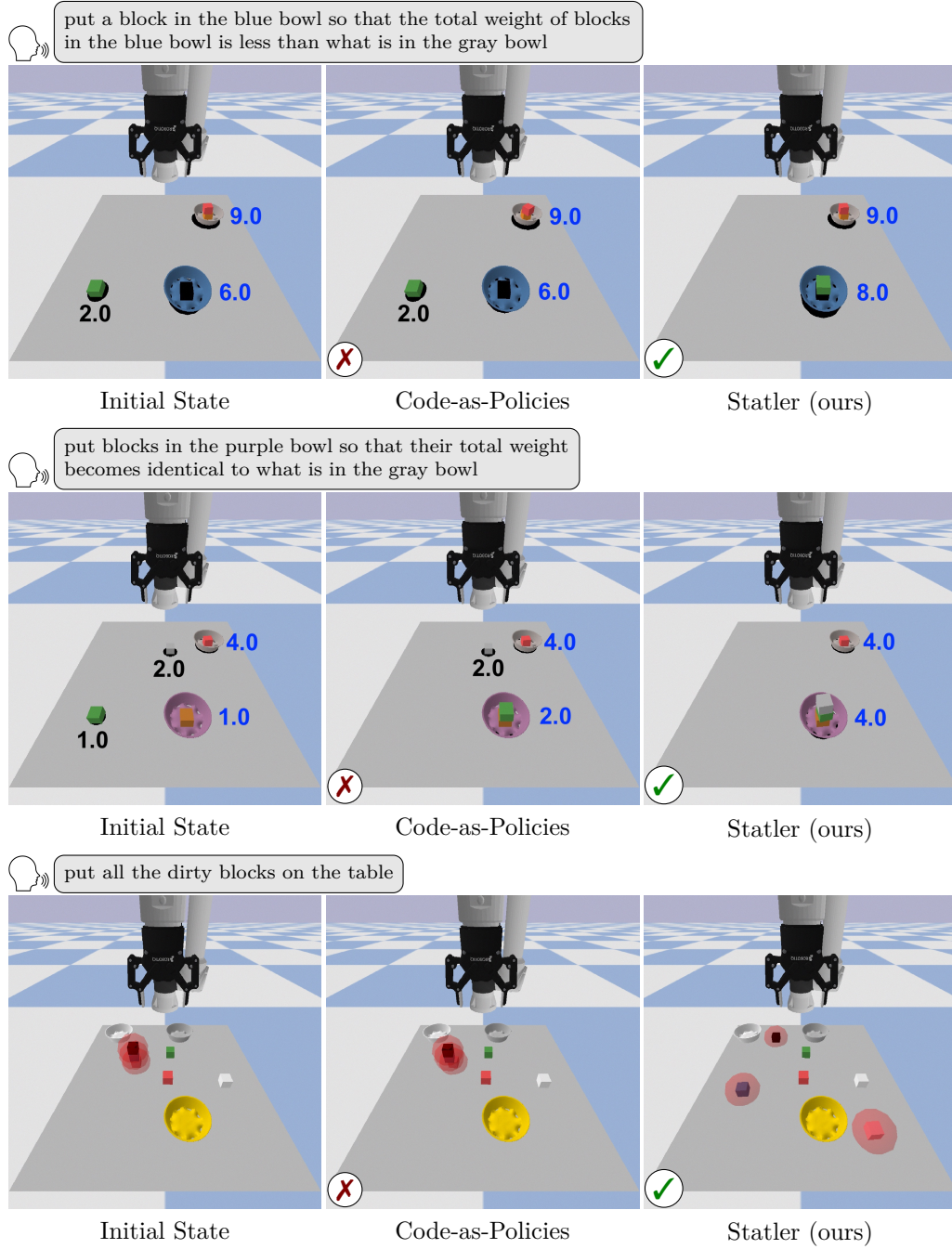


Figure 4.9: Simulation scenarios comparing Code-as-Policies and our state-maintaining Statler on executing natural language instructions.

|              | Non-temporal   |                | Temporal    |                     |
|--------------|----------------|----------------|-------------|---------------------|
|              | CaP            | Statler (ours) | CaP         | Statler (ours)      |
| Pick & Place | 1.00 (62/62)   | 1.00 (68/68)   | 0.31 (9/29) | <b>0.83 (48/58)</b> |
| Disinfection | 0.99 (148/149) | 0.98 (164/168) | 0.05 (1/20) | <b>0.65 (15/23)</b> |
| Weight       | 1.00 (107/107) | 1.00 (107/107) | 0.00 (0/20) | <b>0.55 (11/20)</b> |

Table 4.5: Success rates of Code-as-Policies (CaP) and Statler for non-temporal and temporal queries.

### Analysis of Temporal vs. Non-Temporal Queries

To better understand where Statler excels, we categorize each query as either *temporal* (requiring memory of past interactions) or *non-temporal*. Table 4.5 analyzes the success rates on these query types. Both Statler and the baselines perform near-perfectly on non-temporal queries. However, for temporal queries, the performance of Code-as-Policies drops dramatically, while Statler maintains a high success rate. This clearly indicates that the performance gap is driven by Statler’s superior ability to reason about a consistently updated world state over time.

### State-Maintenance Ablations

To analyze the design of our reader-writer architecture, we compare Statler to three variations:

- **Statler-Unified:** Uses a single LLM for both reading the state and writing updates.
- **Statler-Auto:** Provides the writer with a natural language instruction on how to update the state, rather than few-shot examples.
- **Statler w/o State:** Removes state maintenance entirely, reducing the model to Code-as-Policies.

Table 4.6 shows that removing state maintenance entirely (Statler w/o State) leads to a complete failure on long-horizon tasks. Our proposed reader-writer architecture (Statler) consistently outperforms the unified model, suggesting that specializing the prompts for reading and writing is beneficial. Interestingly, the Statler-Auto variant performs very well, indicating that LLMs can

|                   | Pick & Place       | Disinfection       | Weight             |
|-------------------|--------------------|--------------------|--------------------|
| Statler w/o State | 0.00 (0.54)        | 0.00 (0.68)        | 0.00 (0.84)        |
| Statler-Unified   | 0.40 (0.85)        | 0.35 (0.79)        | 0.50 (0.92)        |
| Statler-Auto      | <b>0.75 (0.88)</b> | <b>0.45 (0.82)</b> | 0.40 (0.90)        |
| Statler (ours)    | 0.50 (0.88)        | 0.40 (0.82)        | <b>0.55 (0.93)</b> |

Table 4.6: Ablation episode (individual step) success rates.

effectively follow abstract instructions for state updates, potentially reducing the need for extensive prompt engineering.

### 4.4.3 Application to Real-World Robotic Manipulation

Finally, to assess the practical utility of each system, we deployed Transcrib3D and Statler on a physical UR5 robotic arm for pick-and-place tasks involving complex language and state-dependent logic. Rather than jointly deploying both systems as a single pipeline, we evaluate their individual contributions on a shared platform: Section 4.4.3 isolates Transcrib3D’s impact on language grounding by pairing it with the standard Code-as-Policies execution backbone, while Section 4.4.3 isolates Statler’s impact on state-dependent execution using Code-as-Policies-grounded perception. We discuss a concrete path toward jointly deploying the two systems in Section 4.5.3.

#### Referring Expressions for Manipulation

We first evaluated the impact of using Transcrib3D for perception within the Code-as-Policies framework. We tasked the robot with instructions containing complex referring expressions. As shown in Figures 4.4 and 4.10, Transcrib3D enables the robot to correctly ground phrases like “the toy duckie surrounded by the cups” and “the black cup farthest from the shortest cup,” which a standard perception module fails to interpret. Our method (CaP+Transcrib3D) achieved an 80% instruction success rate, compared to just 20% for the baseline (CaP+MDETR), demonstrating the critical role of robust language grounding for real-world tasks.



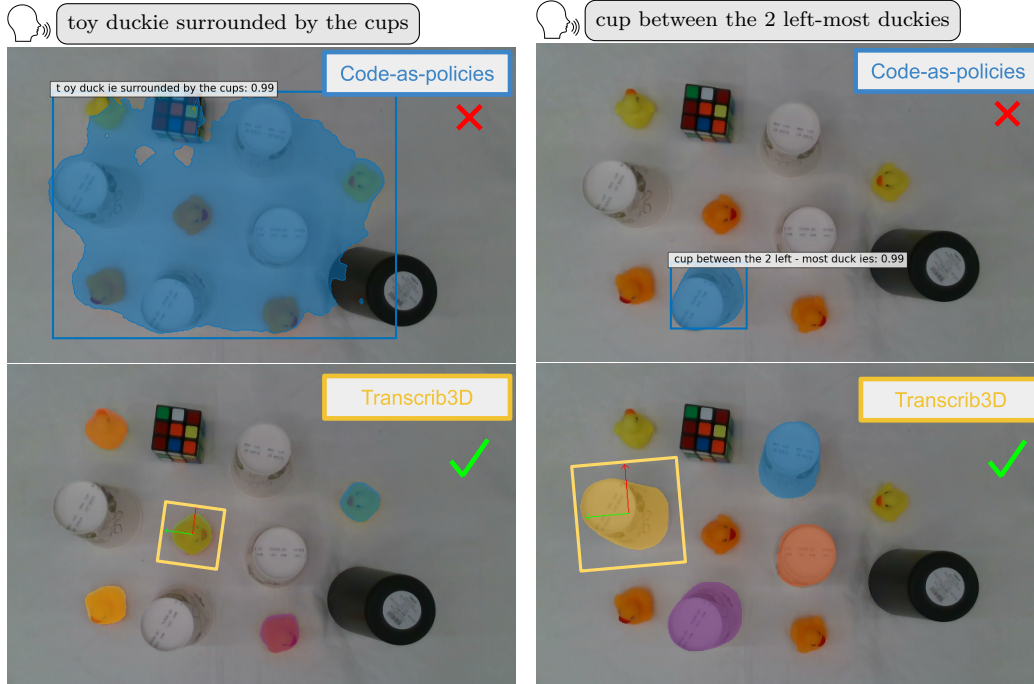


Figure 4.10: Qualitative comparisons of the grounding performance of CaP+MDETR (top) and CaP+Transcrib3D (bottom).

### State-Maintaining Reasoning in the Real World

We then evaluated Statler’s ability to handle state-dependent logic on the real robot. The task required the robot to reason about object occlusion. For instance, after placing a cup over a block, it must remember that the block is covered and remove the cup before it can manipulate the block again. Figure 4.11 shows a scenario where Code-as-Policies fails because it loses track of the covered block, while Statler correctly maintains the state and successfully completes the multi-step instruction. Over 10 episodes, Statler achieved a 40% episode success rate, compared to 20% for Code-as-Policies, confirming that the benefits of explicit state maintenance translate directly to improved performance on a physical robot.

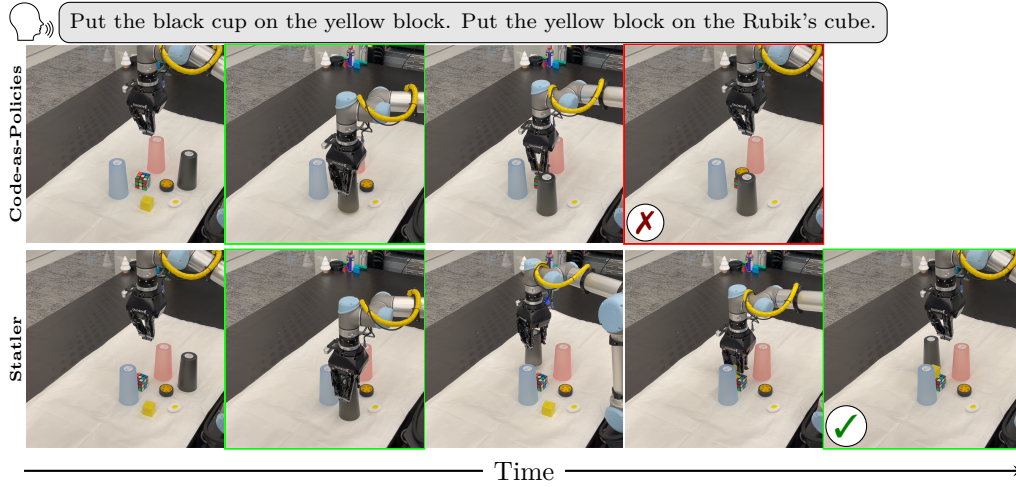


Figure 4.11: A real robot experiment comparing Code-as-Policies and Statler. The instruction requires remembering that the yellow block is under the black cup. Code-as-Policies loses track of this and manipulates the wrong object. Statler maintains the state, removes the cup, and completes the task.

## 4.5 Discussion and Limitations

Despite the significant advances presented in this chapter, our framework inherits certain limitations from its core components, which in turn illuminate promising directions for future research. We first discuss the limitations of Transcrib3D and Statler in turn, and then describe how the two systems could be brought together into a unified end-to-end pipeline.

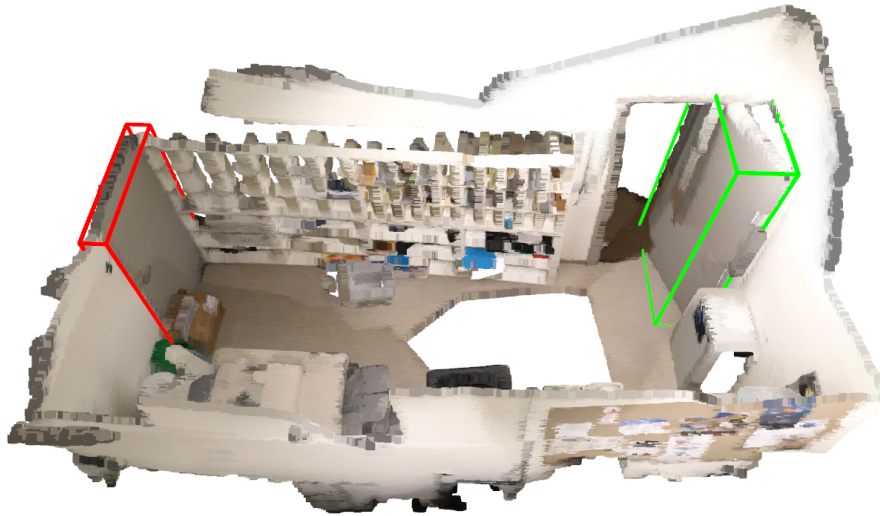
### 4.5.1 Limitations of Transcrib3D

A primary limitation of Transcrib3D is the information loss that occurs during the scene transcription process. By abstracting objects to their bounding boxes, colors, and categories, we lose fine-grained geometric and textural details. This can lead to failures when referring expressions rely on nuanced shape attributes (e.g., “the cylinder shaped trash can”) or functional states (e.g., an “open” door) that are not captured in the transcript, as illustrated in Figure 4.12. The performance of Transcrib3D is also fundamentally bounded by the quality of the upstream 3D object detector; improvements in 3D perception will directly translate to more robust language grounding.





(a) Utterance: “The cylinder shaped trash can.” Our model is unable to ground the “cylinder” shape, which is absent from the transcript.



(b) Utterance: “This door is open and has nothing blocking it.” This query requires understanding the concept of “open”, which is hard to infer from a bounding box.

Figure 4.12: Typical failure cases in Transcrib3D, with ground-truth bounding boxes shown in green and our predictions in red. These examples highlight the information bottleneck of using simplified textual descriptions.

### 4.5.2 Limitations of Statler

Statler’s reasoning is constrained by the inherent weaknesses of current LLMs. While explicit state maintenance mitigates memory degradation, the model can still suffer from logical errors or hallucinations, such as incorrectly updating a latent property or misinterpreting complex weight-based constraints. These issues underscore that while LLMs are powerful reasoning engines, they are not infallible symbolic planners. Future work could explore hybrid approaches that combine the flexibility of LLMs with the formal guarantees of traditional planners.

### 4.5.3 Pathways to Integration

Although we developed and evaluated Transcrib3D and Statler independently, the two systems address genuinely complementary problems—grounding *what* the user means within the 3D scene and tracking *how* the world evolves under the robot’s actions—and their designs admit a natural path to joint deployment. We sketch here how such an end-to-end pipeline could be constructed, what obstacles would need to be overcome, and why the conceptual coupling is worth pursuing despite the absence of an integrated implementation in the present work.

**A concrete integration path.** At the start of an episode, Transcrib3D would consume the colored point-cloud of the scene and produce its object-centric scene transcript, as described in Section 4.3.1. The objects it identifies, together with their categories, positions, spatial extents, and colors, would then be used to seed Statler’s initial world-state representation rather than relying on a hand-specified JSON state. When the user issues a command, any referring expressions within it would be routed through Transcrib3D’s iterative code-generation module (Section 4.3.1) to yield a concrete object identifier, which the Statler reader would then consume when generating manipulation code. After each action, the Statler writer would update the world state in its usual JSON-like format. In most cases this update is sufficient on its own: the writer already tracks which objects have moved and how their spatial relationships have changed. A targeted re-transcription by Transcrib3D would be reserved for situations in which the writer’s text-only update is unreliable—specifically, when a manipulation may have revealed previously occluded objects whose existence is absent from the current tran-

script. Conversely, when an action occludes an object (e.g., placing a cup over a block), Transcrib3D’s perception would no longer detect it, making the writer’s state update essential for maintaining a complete picture of the scene. In this arrangement, Transcrib3D functions as the grounding oracle and Statler as the stateful controller, with their exchanges mediated entirely through text.

**Remaining challenges.** Realizing this design raises several non-trivial issues. First, the representations at the two interface boundaries are not drop-in compatible: Transcrib3D’s transcript is a flat list of detected objects with numeric attributes, while Statler’s state is a structured, task-relevant JSON-like object whose schema is shaped by few-shot examples, so a schema-bridging layer would be needed to translate between them faithfully. Second, although the selective re-transcription strategy described above avoids the cost of running Mask3D after every action, it introduces a detection problem of its own: the system must determine when a manipulation may have revealed previously occluded objects, which itself requires reasoning about occlusion geometry that neither system currently performs. Third, integration amplifies the impact of Transcrib3D’s existing perception limitations (discussed in Section 4.5.1). In standalone operation, a detection error—such as misclassifying an object or missing one entirely—produces a single incorrect grounding result. In the integrated system, the same error would be embedded in Statler’s persistent state and propagate through every subsequent action and state update for the remainder of the episode, since the writer has no independent perception to detect or correct the discrepancy.

**Why the conceptual coupling remains meaningful.** Even without a joint implementation, the conceptual connection between Transcrib3D and Statler is load-bearing. They target the two essential capabilities that any language-driven embodied agent must possess, and both build on the Code-as-Policies paradigm, which provides a common substrate on which each system’s capabilities can be exposed to the other as callable functions (their generated code differs in purpose—Transcrib3D’s performs spatial arithmetic over detected objects while Statler’s invokes manipulation primitives—but both fit the same LLM-as-code-generator interface). Moreover, both systems use text as the primary medium for LLM reasoning, which makes their representations naturally composable: a transcript can seed a state, and a

state can be re-queried via a referring expression. We view the integration sketched above as a natural next step rather than a speculative leap, and present it here to make concrete the conceptual bridge between perception and action that this chapter has argued for throughout.

#### 4.5.4 Relation to Subsequent Work

**Transcrib3D.** The two years since Transcrib3D’s publication have seen a proliferation of 3D-aware language models that ingest raw 3D inputs directly rather than going through a textual transcript. Alongside the already-cited Grounded 3D-LLM [134], 3D-GRAND [159], and LLM-Grounder [160], the subsequent wave includes LLaVA-3D [164], Chat-Scene [142], Spatial-RGPT [135], and SceneFun3D [136]. These methods narrow Transcrib3D’s principal limitation—the information loss at the transcript bottleneck on shape attributes and functional states—by consuming raw 3D geometry and imagery. Transcrib3D’s continuing value lies in a more specific regime. First, its performance remains competitive despite the bottleneck, supporting the thesis that text is a sufficient unifying medium when the reasoning burden is high. Second, the design composes with off-the-shelf 3D detectors and does not require the 3D-grounded annotations whose scarcity motivated the method in the first place, making it well-suited to data-sparse deployment in which end-to-end 3D-VLMs are hard to train or adapt.

**Statler.** Two axes have developed rapidly since Statler’s publication. On one, end-to-end vision-language-action (VLA) models—RT-2 [87], Open X-Embodiment [152], Octo [140], and OpenVLA [147]—pursue state as a learned quantity implicit in a single network rather than as an explicit textual object. On the other, explicit-state LLM agents such as StateAct [153] and RoboHorizon [166], together with structured reasoning techniques [125, 130, 154], extend the reader-writer idea in different directions. Statler’s continuing value is the clean separation of world-state writing from action selection, and the empirical evidence that *explicit textual state* yields robustness on state-dependent tasks that end-to-end VLAs have not matched. As VLA models mature, the reader-writer pattern remains a sensible symbolic scaffold on top of a learned policy rather than a competitor to it.

## 4.6 Conclusion

This chapter bridges the gap between the spatial and kinematic layers of our world model, established in the preceding chapters, and the execution of intelligent, goal-directed robotic action. We demonstrated that by harnessing the reasoning capabilities of Large Language Models, a robotic agent can effectively interpret complex human language and maintain a consistent understanding of its environment over long-horizon tasks. This was accomplished through two complementary systems, Transcrib3D and Statler, which together form a framework for LLM-powered embodied reasoning.

First, we introduced Transcrib3D, a novel framework that tackles the critical challenge of grounding 3D referring expressions. By pioneering the use of *text* as a unifying medium, Transcrib3D sidesteps the data scarcity problem inherent in traditional multi-modal approaches. It first transcribes a 3D scene into an object-centric textual description and then leverages an LLM’s sophisticated linguistic and spatial reasoning abilities—augmented by iterative code generation—to resolve complex, ambiguous, and relational user commands. Our experiments on the ReferIt3D and ScanRefer benchmarks show that this approach achieves state-of-the-art performance, significantly outperforming prior methods.

Second, we presented Statler, a state-maintaining architecture designed to overcome the limitations of LLMs in long-horizon planning. Standard LLM-based agents, which operate in a model-free manner, often fail to track changes in the world, leading to a collapse in performance on multi-step, state-dependent tasks. Statler addresses this by explicitly maintaining a representation of the world state through its integral *world-state reader* and *world-state writer*. Evaluations on challenging simulated and real-world manipulation tasks revealed that Statler dramatically outperforms contemporary model-free methods, especially on queries that require reasoning over past interactions.

Taken together, Transcrib3D and Statler address the two complementary halves of the journey from high-level human intent to successful robotic execution. Transcrib3D targets the interpretive front-end, accurately grounding a user’s goal within the 3D scene, while Statler targets the stateful execution back-end, enabling the robot to carry out complex, sequential plans. We developed and evaluated the two systems independently rather than as a single jointly deployed pipeline; the successful deployment of each on a physical robot, as shown in Figures 4.4 and 4.11, validates their individual practical

utility, and Section 4.5.3 sketches a concrete path toward combining them into a unified end-to-end system.

Ultimately, the success of both systems points to a critical insight: the primary bottleneck for current multi-modal embodied intelligence may not be the lack of a unified vision-language representation, but rather the scarcity of annotated 3D data and the limitations of the reasoning modules themselves. Our work demonstrates that by decomposing the problem and leveraging powerful, pre-existing LLM reasoners, we can achieve remarkable performance. This suggests a path forward focused on creating richer scene representations and more robust, verifiable reasoning mechanisms, paving the way for the next generation of embodied AI.

# Chapter 5

## Conclusion

This thesis has confronted the multifaceted challenge of enabling robotic agents to perceive, reason about, and intelligently act within large-scale, dynamic, and human-centric 3D environments. We have argued that true embodied intelligence requires a holistic approach, one that builds a comprehensive world model from the ground up and then connects it to high-level, language-driven reasoning. To this end, we have presented a unified framework that progresses logically from scalable scene reconstruction to the understanding of dynamic objects, and finally to the execution of complex, long-horizon tasks guided by natural language. The research presented herein makes a series of core contributions that, together, form a complete pipeline from pixels to actions, advancing the state of the art in robotic scene understanding and embodied reasoning.

### 5.1 Summary of Contributions

The journey toward creating autonomous and collaborative robots begins with a robust and scalable representation of the physical world. Our work has systematically addressed the key components required to build and leverage such a representation.

First, in Chapter 2, we tackled the fundamental problem of **spatial scalability in 3D scene reconstruction**. We introduced **NeRFuser**, a novel framework for composing large-scale, high-fidelity scenes from multiple, independently trained Neural Radiance Fields [137]. By developing a *registration from re-rendering* technique that operates without access to original training



data, we provided a practical solution for creating expansive 3D scene representations in decentralized or multi-agent scenarios. Complemented by a fine-grained, *sample-based blending* algorithm, NeRFuser equips a robot with comprehensive spatial awareness of its environment, providing the foundation for all subsequent reasoning.

Second, in Chapter 3, we enriched this static scene representation with a crucial layer of **temporal and functional understanding**. We presented **SplArt**, a self-supervised method for reconstructing articulated objects and inferring their kinematic properties from sparse observations of just two end states [171]. By augmenting 3D Gaussian Splatting [104] with a differentiable mobility parameter and a multi-stage optimization strategy, SPLART recovers an object’s articulation model without 3D supervision or category-specific priors. This contribution moves beyond static geometry, equipping the agent with the ability to perceive *how* objects can move and be manipulated—a critical prerequisite for meaningful physical interaction.

Finally, in Chapter 4, we bridged the gap between the spatial and kinematic layers of our world model and intelligent action by harnessing the power of **Large Language Models for embodied reasoning**. We introduced two complementary systems that serve as a cognitive engine for the robot. **Transcrib3D** addresses the challenge of language grounding by reasoning over a textual transcript of the 3D scene, enabling the robot to robustly interpret complex, free-form referring expressions [138]. Concurrently, **Statler** tackles long-horizon planning with a state-maintaining architecture that tracks the world state over time [162]. This lets the robot execute multi-step, state-dependent tasks by reasoning over both user intent and a consistently updated world state. Together, these systems translate high-level human goals into physically-grounded robotic behavior.

## 5.2 Broader Implications

Viewed in concert, these contributions form a cohesive and powerful pipeline that charts a clear path from perception to action. The overarching thesis is that by systematically building a rich, multi-layered world model and then coupling it with a powerful reasoning engine, we can create robotic agents that are significantly more capable, flexible, and collaborative. The implications of this work are threefold.

First, this thesis champions a **scalable and compositional founda-**



**tion for real-world perception.** The success of NeRFuser demonstrates a practical path toward building the expansive 3D scene representations necessary for robots operating outside of constrained laboratory settings. By enabling the fusion of independently captured sub-maps without original data, our work supports decentralized, multi-agent, and lifelong learning scenarios. This establishes a critical principle: a robot’s spatial awareness need not be monolithic but can be dynamically constructed and expanded over time, scaling naturally with the environments the robot must operate in.

Second, our work illustrates the pivotal transition from **static geometric models to dynamic functional understanding**. The contribution of SPLART moves beyond simply representing what a scene looks like, and toward understanding what it affords. By inferring the kinematics of articulated objects from sparse observations, we equip the agent with a model of physical possibility—the knowledge of how objects can be manipulated. This shift from a descriptive to a functional world model is a crucial step in bridging the significant gap between passive perception and active, purposeful physical interaction.

Third, we demonstrate a **principled framework for connecting rich world models to high-level reasoning**. The systems presented in the final chapter, Transcrib3D and Statler, are not standalone reasoners; their success is contingent upon the detailed spatial and kinematic layers of our world model established by the preceding work. This highlights a key implication: the power of large language models for robotics is most effectively unlocked when they can ground their reasoning in a rich, accurate, and dynamic representation of the physical world. Our work provides a blueprint for this integration, showing how a robust perceptual pipeline can serve as the necessary foundation for sophisticated language understanding and long-horizon, stateful planning.

Collectively, these principles pave the way for robotic systems that can be deployed in unstructured, real-world environments, collaborate with humans through natural language, and perform complex, multi-step tasks that were previously intractable.

### 5.3 Future Directions

While this thesis provides a comprehensive framework, it also opens up numerous exciting avenues for future research. The limitations of the individual

components and the potential for deeper integration suggest several promising directions for extending this work.

- **Active and Dynamic Large-Scale Mapping:** The current NeRFuser framework operates on pre-existing, static NeRFs. A natural extension would be to handle dynamic scenes, disentangling static background from transient objects during registration and blending. Furthermore, this could be integrated into an active SLAM pipeline, where a robot uses the composite map to strategically decide where to explore next, capturing new NeRF sub-maps to incrementally build a large-scale, ever-evolving 3D scene representation.
- **Toward More Complex Dynamic Objects:** The current work focuses on single-DoF, two-part articulations. Another important direction is to handle objects with more complex, multi-part kinematic chains and topologies. Recent works like ArticulatedGS [169] have begun to explore this frontier, yet robustly handling arbitrary topologies in the wild remains an open challenge. This would require developing more sophisticated methods for part segmentation and association, potentially incorporating weak semantic priors or learning from interaction to disentangle complex motions. Furthermore, extending the framework to model non-rigid deformations and other dynamic phenomena would lead to an even richer and more complete world model.
- **Richer Scene Representations and Grounding:** The scene transcription in Transcrib3D, while effective, abstracts away fine-grained geometric and textural details. Future work could explore creating richer, more descriptive transcripts that capture shape, material properties, and functional affordances. Emerging approaches such as Grounded 3D-LLM [134] suggest that unified models are a viable path, though bridging the gap between text-based reasoning and raw 3D inputs remains a key area for innovation. This could involve leveraging Vision-Language Models (VLMs) to generate more nuanced descriptions, thereby enabling the grounding of an even wider and more complex range of human language.
- **Hybrid Reasoning for Planning and Verification:** While Statler demonstrates the power of LLMs for stateful reasoning, these models

can still suffer from logical errors or hallucinations. A promising direction is to explore hybrid reasoning architectures that combine the flexibility and common-sense knowledge of LLMs with the formal guarantees of traditional symbolic planners. An LLM could be used to propose high-level plans or interpret ambiguous goals, while a symbolic verifier ensures the logical consistency and ultimate safety of the resulting actions.

- **Lifelong Learning and Adaptation:** The current framework operates on a per-task basis. A truly autonomous agent should be able to learn and adapt continuously from its experiences. Future research could focus on integrating these components into a lifelong learning system, where the robot constantly updates its large-scale map (NeRFuser), discovers and models new articulated objects (SPLART), and refines its understanding of language and task execution (Transcrib3D and Statler) through ongoing interaction with its environment and human collaborators.

## 5.4 Concluding Remarks

The central goal of this thesis was to develop a scalable and cohesive framework for 3D scene understanding and embodied reasoning, enabling robots to intelligently interact with the world based on high-level human instruction. By building from a foundation of scalable spatial mapping, adding a layer of dynamic object understanding, and crowning the system with a powerful language-driven reasoning engine, we have presented a complete pipeline that translates raw sensory input into meaningful, goal-directed action. The collective contributions of NeRFuser, SPLART, Transcrib3D, and Statler represent a significant step toward the creation of more autonomous, capable, and collaborative robotic systems, paving the way for a future where robots can seamlessly perceive, understand, and operate within the rich complexities of the human world.

# Bibliography

- [1] S. Rusinkiewicz and M. Levoy. “Efficient variants of the ICP algorithm”. In: *International Conference on 3-D Digital Imaging and Modeling (3DIM)*. IM-01. IEEE Comput. Soc, 2001, pp. 145–152. DOI: [10.1109/im.2001.924423](https://doi.org/10.1109/im.2001.924423).
- [2] Zhou Wang et al. “Image quality assessment: from error visibility to structural similarity”. In: *IEEE Transactions on Image Processing (TIP)* 13.4 (Apr. 2004), pp. 600–612. ISSN: 1941-0042. DOI: [10.1109/tip.2003.819861](https://doi.org/10.1109/tip.2003.819861).
- [3] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. “Fast Point Feature Histograms (FPFH) for 3D registration”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2009. DOI: [10.1109/robot.2009.5152473](https://doi.org/10.1109/robot.2009.5152473).
- [4] Christopher B. Choy et al. “3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016, pp. 628–644. ISBN: 9783319464848. DOI: [10.1007/978-3-319-46484-8\\_38](https://doi.org/10.1007/978-3-319-46484-8_38).
- [5] Johannes L. Schonberger and Jan-Michael Frahm. “Structure-from-Motion Revisited”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2016. DOI: [10.1109/cvpr.2016.445](https://doi.org/10.1109/cvpr.2016.445).
- [6] Johannes L. Schönberger et al. “Pixelwise View Selection for Unstructured Multi-View Stereo”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016, pp. 501–518. ISBN: 9783319464879. DOI: [10.1007/978-3-319-46487-9\\_31](https://doi.org/10.1007/978-3-319-46487-9_31).

- [7] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. “Fast Global Registration”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2016, pp. 766–782. ISBN: 9783319464756. DOI: [10.1007/978-3-319-46475-6\\_47](https://doi.org/10.1007/978-3-319-46475-6_47).
- [8] Angela Dai et al. “ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017, pp. 2432–2443. DOI: [10.1109/cvpr.2017.261](https://doi.org/10.1109/cvpr.2017.261).
- [9] Haoqiang Fan, Hao Su, and Leonidas Guibas. “A Point Set Generation Network for 3D Object Reconstruction from a Single Image”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017, pp. 2463–2471. DOI: [10.1109/cvpr.2017.264](https://doi.org/10.1109/cvpr.2017.264).
- [10] Ruizhen Hu et al. “Learning to predict part mobility from a single static snapshot”. In: *ACM Transactions on Graphics (TOG)* 36.6 (Nov. 2017), pp. 1–13. ISSN: 1557-7368. DOI: [10.1145/3130800.3130811](https://doi.org/10.1145/3130800.3130811).
- [11] Ashish Vaswani et al. “Attention Is All You Need”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2017. DOI: [10.48550/arxiv.1706.03762](https://doi.org/10.48550/arxiv.1706.03762).
- [12] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. “SuperPoint: Self-Supervised Interest Point Detection and Description”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, June 2018. DOI: [10.1109/cvprw.2018.00060](https://doi.org/10.1109/cvprw.2018.00060).
- [13] Li Yi et al. “Deep part induction from articulated object pairs”. In: *ACM Transactions on Graphics (TOG)* 37.6 (Dec. 2018), pp. 1–15. ISSN: 1557-7368. DOI: [10.1145/3272127.3275027](https://doi.org/10.1145/3272127.3275027).
- [14] Richard Zhang et al. “The Unreasonable Effectiveness of Deep Features as a Perceptual Metric”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2018. DOI: [10.1109/cvpr.2018.00068](https://doi.org/10.1109/cvpr.2018.00068).

- [15] Mihai Dusmanu et al. “D2-Net: A Trainable CNN for Joint Description and Detection of Local Features”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019, pp. 8084–8093. DOI: [10.1109/cvpr.2019.00828](https://doi.org/10.1109/cvpr.2019.00828).
- [16] Lars Mescheder et al. “Occupancy Networks: Learning 3D Reconstruction in Function Space”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019. DOI: [10.1109/cvpr.2019.00459](https://doi.org/10.1109/cvpr.2019.00459).
- [17] Kaichun Mo et al. “PartNet: A Large-Scale Benchmark for Fine-Grained and Hierarchical Part-Level 3D Object Understanding”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019, pp. 909–918. DOI: [10.1109/cvpr.2019.00100](https://doi.org/10.1109/cvpr.2019.00100).
- [18] Jeong Joon Park et al. “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019, pp. 165–174. DOI: [10.1109/cvpr.2019.00025](https://doi.org/10.1109/cvpr.2019.00025).
- [19] Siddharth Patki et al. “Inferring Compact Representations for Efficient Natural Language Understanding of Robot Instructions”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2019, pp. 6926–6933. DOI: [10.1109/icra.2019.8793667](https://doi.org/10.1109/icra.2019.8793667).
- [20] Jerome Revaud et al. *R2D2: Repeatable and Reliable Detector and Descriptor*. 2019. DOI: [10.48550/arxiv.1906.06195](https://doi.org/10.48550/arxiv.1906.06195).
- [21] Paul-Edouard Sarlin et al. “From Coarse to Fine: Robust Hierarchical Localization at Large Scale”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019. DOI: [10.1109/cvpr.2019.01300](https://doi.org/10.1109/cvpr.2019.01300).
- [22] Xiaogang Wang et al. “Shape2Motion: Joint Analysis of Motion Parts and Attributes From 3D Shapes”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2019, pp. 8868–8876. DOI: [10.1109/cvpr.2019.00908](https://doi.org/10.1109/cvpr.2019.00908).
- [23] Zihao Yan et al. “RPM-Net: recurrent prediction of motion and parts from point cloud”. In: *ACM Transactions on Graphics (TOG)* 38.6 (Nov. 2019), pp. 1–15. ISSN: 1557-7368. DOI: [10.1145/3355089.3356573](https://doi.org/10.1145/3355089.3356573).

- [24] Panos Achlioptas et al. “ReferIt3D: Neural Listeners for Fine-Grained 3D Object Identification in Real-World Scenes”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2020, pp. 422–440. ISBN: 9783030584528. DOI: [10.1007/978-3-030-58452-8\\_25](https://doi.org/10.1007/978-3-030-58452-8_25).
- [25] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. DOI: [10.48550/arxiv.2005.14165](https://doi.org/10.48550/arxiv.2005.14165).
- [26] Dave Zhenyu Chen, Angel X. Chang, and Matthias Nießner. “Scan-Refer: 3D Object Localization in RGB-D Scans Using Natural Language”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2020, pp. 202–221. ISBN: 9783030585655. DOI: [10.1007/978-3-030-58565-5\\_13](https://doi.org/10.1007/978-3-030-58565-5_13).
- [27] Li Jiang et al. “PointGroup: Dual-Set Point Grouping for 3D Instance Segmentation”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020. DOI: [10.1109/cvpr42600.2020.00492](https://doi.org/10.1109/cvpr42600.2020.00492).
- [28] Xiaolong Li et al. “Category-Level Articulated Object Pose Estimation”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020, pp. 3703–3712. DOI: [10.1109/cvpr42600.2020.00376](https://doi.org/10.1109/cvpr42600.2020.00376).
- [29] Arjun Majumdar et al. “Improving Vision-and-Language Navigation with Image-Text Pairs from the Web”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2020, pp. 259–274. ISBN: 9783030585396. DOI: [10.1007/978-3-030-58539-6\\_16](https://doi.org/10.1007/978-3-030-58539-6_16).
- [30] Ben Mildenhall et al. “NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis”. In: *European Conference on Computer Vision (ECCV)*. Springer International Publishing, 2020, pp. 405–421. ISBN: 9783030584528. DOI: [10.1007/978-3-030-58452-8\\_24](https://doi.org/10.1007/978-3-030-58452-8_24).
- [31] Paul-Edouard Sarlin et al. “SuperGlue: Learning Feature Matching With Graph Neural Networks”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020, pp. 4937–4946. DOI: [10.1109/cvpr42600.2020.00499](https://doi.org/10.1109/cvpr42600.2020.00499).

- [32] Fanbo Xiang et al. “SAPIEN: A SimulAted Part-Based Interactive ENvironment”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2020, pp. 11094–11104. DOI: [10.1109/cvpr42600.2020.01111](https://doi.org/10.1109/cvpr42600.2020.01111).
- [33] Jonathan T. Barron et al. “Mip-NeRF: A Multiscale Representation for Anti-Aliasing Neural Radiance Fields”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021. DOI: [10.1109/iccv48922.2021.00580](https://doi.org/10.1109/iccv48922.2021.00580).
- [34] Mark Chen et al. *Evaluating Large Language Models Trained on Code*. 2021. DOI: [10.48550/arxiv.2107.03374](https://doi.org/10.48550/arxiv.2107.03374).
- [35] Samir Yitzhak Gadre, Kiana Ehsani, and Shuran Song. “Act the Part: Learning Interaction Strategies for Articulated Object Part Discovery”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021, pp. 15732–15741. DOI: [10.1109/iccv48922.2021.01546](https://doi.org/10.1109/iccv48922.2021.01546).
- [36] Dan Hendrycks et al. *Measuring Coding Challenge Competence With APPS*. 2021. DOI: [10.48550/arxiv.2105.09938](https://doi.org/10.48550/arxiv.2105.09938).
- [37] Yoonwoo Jeong et al. “Self-Calibrating Neural Radiance Fields”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021, pp. 5826–5834. DOI: [10.1109/iccv48922.2021.00579](https://doi.org/10.1109/iccv48922.2021.00579).
- [38] Chen-Hsuan Lin et al. “BARF: Bundle-Adjusting Neural Radiance Fields”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021, pp. 5721–5731. DOI: [10.1109/iccv48922.2021.00569](https://doi.org/10.1109/iccv48922.2021.00569).
- [39] Ben Mildenhall et al. “NeRF: representing scenes as neural radiance fields for view synthesis”. In: *Communications of the ACM (CACM)*. 65.1 (Dec. 2021), pp. 99–106. ISSN: 1557-7317. DOI: [10.1145/3503250](https://doi.org/10.1145/3503250).
- [40] Jiteng Mu et al. “A-SDF: Learning Disentangled Signed Distance Functions for Articulated Shape Representation”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021, pp. 12981–12991. DOI: [10.1109/iccv48922.2021.01276](https://doi.org/10.1109/iccv48922.2021.01276).
- [41] Maxwell Nye et al. *Show Your Work: Scratchpads for Intermediate Computation with Language Models*. 2021. DOI: [10.48550/arxiv.2112.00114](https://doi.org/10.48550/arxiv.2112.00114).



- [42] Michael Oechsle, Songyou Peng, and Andreas Geiger. “UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021. DOI: [10.1109/iccv48922.2021.00554](https://doi.org/10.1109/iccv48922.2021.00554).
- [43] Christian Reiser et al. “KiloNeRF: Speeding up Neural Radiance Fields with Thousands of Tiny MLPs”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021. DOI: [10.1109/iccv48922.2021.01407](https://doi.org/10.1109/iccv48922.2021.01407).
- [44] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. *CLIPort: What and Where Pathways for Robotic Manipulation*. 2021. DOI: [10.48550/arxiv.2109.12098](https://doi.org/10.48550/arxiv.2109.12098).
- [45] Edgar Sucar et al. “iMAP: Implicit Mapping and Positioning in Real-Time”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021, pp. 6209–6218. DOI: [10.1109/iccv48922.2021.00617](https://doi.org/10.1109/iccv48922.2021.00617).
- [46] Towaki Takikawa et al. “Neural Geometric Level of Detail: Real-time Rendering with Implicit 3D Shapes”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2021, pp. 11353–11362. DOI: [10.1109/cvpr46437.2021.01120](https://doi.org/10.1109/cvpr46437.2021.01120).
- [47] Peng Wang et al. “NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2021, pp. 27171–27183. DOI: [10.48550/arxiv.2106.10689](https://doi.org/10.48550/arxiv.2106.10689).
- [48] Zirui Wang et al. *NeRF-: Neural Radiance Fields Without Known Camera Parameters*. 2021. DOI: [10.48550/arxiv.2102.07064](https://doi.org/10.48550/arxiv.2102.07064).
- [49] Zhengyuan Yang et al. “SAT: 2D Semantics Assisted Training for 3D Visual Grounding”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2021. DOI: [10.1109/iccv48922.2021.00187](https://doi.org/10.1109/iccv48922.2021.00187).
- [50] Lin Yen-Chen et al. “iNeRF: Inverting Neural Radiance Fields for Pose Estimation”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Sept. 2021. DOI: [10.1109/iros51168.2021.9636708](https://doi.org/10.1109/iros51168.2021.9636708).

- [51] Michal Adamkiewicz et al. “Vision-Only Robot Navigation in a Neural Radiance World”. In: *IEEE Robotics and Automation Letters (RA-L)* 7.2 (Apr. 2022), pp. 4606–4613. ISSN: 2377-3774. DOI: [10.1109/lra.2022.3150497](https://doi.org/10.1109/lra.2022.3150497).
- [52] Michael Ahn et al. “Do As I Can, Not As I Say: Grounding Language in Robotic Affordances”. In: *Conference on Robot Learning (CoRL)*. 2022. DOI: [10.48550/arxiv.2204.01691](https://doi.org/10.48550/arxiv.2204.01691).
- [53] Jonathan T. Barron et al. “Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022. DOI: [10.1109/cvpr52688.2022.00539](https://doi.org/10.1109/cvpr52688.2022.00539).
- [54] Daigang Cai et al. “3DJCG: A Unified Framework for Joint Dense Captioning and Visual Grounding on 3D Point Clouds”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 16443–16452. DOI: [10.1109/cvpr52688.2022.01597](https://doi.org/10.1109/cvpr52688.2022.01597).
- [55] Bei Chen et al. *CodeT: Code Generation with Generated Tests*. 2022. DOI: [10.48550/arxiv.2207.10397](https://doi.org/10.48550/arxiv.2207.10397).
- [56] Dave Zhenyu Chen et al. “D<sup>3</sup>Net: A Unified Speaker-Listener Architecture for 3D Dense Captioning and Visual Grounding”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, 2022, pp. 487–505. ISBN: 9783031198243. DOI: [10.1007/978-3-031-19824-3\\_29](https://doi.org/10.1007/978-3-031-19824-3_29).
- [57] Jiaming Chen et al. *Learning Point-Language Hierarchical Alignment for 3D Visual Grounding*. 2022. DOI: [10.48550/arxiv.2210.12513](https://doi.org/10.48550/arxiv.2210.12513).
- [58] Shizhe Chen et al. *Language Conditioned Spatial Relation Reasoning for 3D Object Grounding*. 2022. DOI: [10.48550/arxiv.2211.09646](https://doi.org/10.48550/arxiv.2211.09646).
- [59] Sara Fridovich-Keil et al. “Plenoxels: Radiance Fields without Neural Networks”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 5491–5500. DOI: [10.1109/cvpr52688.2022.00542](https://doi.org/10.1109/cvpr52688.2022.00542).
- [60] Shijia Huang et al. “Multi-View Transformer for 3D Visual Grounding”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022. DOI: [10.1109/cvpr52688.2022.01508](https://doi.org/10.1109/cvpr52688.2022.01508).

- [61] Wenlong Huang et al. “Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents”. In: *International Conference on Machine Learning (ICML)*. 2022. DOI: [10.48550/arxiv.2201.07207](https://doi.org/10.48550/arxiv.2201.07207).
- [62] Muhammad Zubair Irshad et al. “CenterSnap: Single-Shot Multi-Object 3D Shape Reconstruction and Categorical 6D Pose and Size Estimation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2022, pp. 10632–10640. DOI: [10.1109/icra46639.2022.9811799](https://doi.org/10.1109/icra46639.2022.9811799).
- [63] Muhammad Zubair Irshad et al. “ShAPO: Implicit Representations for Multi-object Shape, Appearance, and Pose Optimization”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, 2022, pp. 275–292. ISBN: 9783031200861. DOI: [10.1007/978-3-031-20086-1\\_16](https://doi.org/10.1007/978-3-031-20086-1_16).
- [64] Ayush Jain et al. “Bottom Up Top Down Detection Transformers for Language Grounding in Images and Point Clouds”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, 2022, pp. 417–433. ISBN: 9783031200595. DOI: [10.1007/978-3-031-20059-5\\_24](https://doi.org/10.1007/978-3-031-20059-5_24).
- [65] Hanxiao Jiang et al. “OPD: Single-View 3D Openable Part Detection”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, 2022, pp. 410–426. ISBN: 9783031198427. DOI: [10.1007/978-3-031-19842-7\\_24](https://doi.org/10.1007/978-3-031-19842-7_24).
- [66] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. “Ditto: Building Digital Twins of Articulated Objects from Interaction”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 5606–5616. DOI: [10.1109/cvpr52688.2022.00553](https://doi.org/10.1109/cvpr52688.2022.00553).
- [67] Yuki Kawana, Yusuke Mukuta, and Tatsuya Harada. “Unsupervised Pose-aware Part Decomposition for Man-Made Articulated Objects”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, 2022, pp. 558–575. ISBN: 9783031200625. DOI: [10.1007/978-3-031-20062-5\\_32](https://doi.org/10.1007/978-3-031-20062-5_32).
- [68] Yujia Li et al. “Competition-Level Code Generation with AlphaCode”. In: *Science* 378 (2022), pp. 1092–1097. DOI: [10.48550/arxiv.2203.07814](https://doi.org/10.48550/arxiv.2203.07814).

- [69] Liu Liu et al. “AKB-48: A Real-World Articulated Object Knowledge Base”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 14789–14798. DOI: [10.1109/cvpr52688.2022.01439](https://doi.org/10.1109/cvpr52688.2022.01439).
- [70] Sourab Mangrulkar et al. *PEFT: State-of-the-art Parameter-Efficient Fine-Tuning methods*. <https://github.com/huggingface/peft>. 2022.
- [71] Arthur Moreau et al. “LENS: Localization Enhanced by NeRF Synthesis”. In: *Conference on Robot Learning (CoRL)*. PMLR. 2022, pp. 1347–1356. DOI: [10.48550/arxiv.2110.06558](https://doi.org/10.48550/arxiv.2110.06558).
- [72] Thomas Müller et al. “Instant neural graphics primitives with a multiresolution hash encoding”. In: *ACM Transactions on Graphics (TOG)* 41.4 (July 2022), pp. 1–15. ISSN: 1557-7368. DOI: [10.1145/3528223.3530127](https://doi.org/10.1145/3528223.3530127).
- [73] Andrew J. Nam et al. *Learning to Reason With Relational Abstractions*. 2022. DOI: [10.48550/arxiv.2210.02615](https://doi.org/10.48550/arxiv.2210.02615).
- [74] Neil Nie et al. *Structure from Action: Learning Interactions for Articulated Object 3D Structure Discovery*. 2022. DOI: [10.48550/arxiv.2207.08997](https://doi.org/10.48550/arxiv.2207.08997).
- [75] Casey Peat et al. *Zero NeRF: Registration with Zero Overlap*. 2022. DOI: [10.48550/arxiv.2211.12544](https://doi.org/10.48550/arxiv.2211.12544).
- [76] Cheng Sun, Min Sun, and Hwann-Tzong Chen. “Direct Voxel Grid Optimization: Super-fast Convergence for Radiance Fields Reconstruction”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022. DOI: [10.1109/cvpr52688.2022.00538](https://doi.org/10.1109/cvpr52688.2022.00538).
- [77] Matthew Tancik et al. “Block-NeRF: Scalable Large Scene Neural View Synthesis”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 8238–8248. DOI: [10.1109/cvpr52688.2022.00807](https://doi.org/10.1109/cvpr52688.2022.00807).
- [78] Fangyin Wei et al. “Self-supervised Neural Articulated Shape and Appearance Models”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 15795–15805. DOI: [10.1109/cvpr52688.2022.01536](https://doi.org/10.1109/cvpr52688.2022.01536).

- [79] Jason Wei et al. “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2022. DOI: [10.48550/arxiv.2201.11903](https://doi.org/10.48550/arxiv.2201.11903).
- [80] Xiuchao Wu et al. “Scalable neural indoor scene rendering”. In: *ACM Transactions on Graphics (TOG)* 41.4 (July 2022), pp. 1–16. ISSN: 1557-7368. DOI: [10.1145/3528223.3530153](https://doi.org/10.1145/3528223.3530153).
- [81] Yuanbo Xiangli et al. “BungeeNeRF: Progressive Neural Radiance Field for Extreme Multi-scale Scene Rendering”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, 2022, pp. 106–122. ISBN: 9783031198243. DOI: [10.1007/978-3-031-19824-3\\_7](https://doi.org/10.1007/978-3-031-19824-3_7).
- [82] Shunyu Yao et al. *ReAct: Synergizing Reasoning and Acting in Language Models*. 2022. DOI: [10.48550/arxiv.2210.03629](https://doi.org/10.48550/arxiv.2210.03629).
- [83] Andy Zeng et al. *Socratic Models: Composing Zero-Shot Multimodal Reasoning with Language*. 2022. DOI: [10.48550/arxiv.2204.00598](https://doi.org/10.48550/arxiv.2204.00598).
- [84] Xiaoshuai Zhang et al. “NeRFusion: Fusing Radiance Fields for Large-Scale Scene Reconstruction”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2022, pp. 5439–5448. DOI: [10.1109/cvpr52688.2022.00537](https://doi.org/10.1109/cvpr52688.2022.00537).
- [85] Anthropic. *Introducing 100K Context Windows*. Accessed: 2023-05-11. May 2023. URL: <https://www.anthropic.com/index/100k-context-windows>.
- [86] Jonathan T. Barron et al. “Zip-NeRF: Anti-Aliased Grid-Based Neural Radiance Fields”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2023. DOI: [10.1109/iccv51070.2023.01804](https://doi.org/10.1109/iccv51070.2023.01804).
- [87] Anthony Brohan et al. “RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control”. In: *Conference on Robot Learning (CoRL)*. 2023. DOI: [10.48550/arxiv.2307.15818](https://doi.org/10.48550/arxiv.2307.15818).
- [88] Yu Chen and Gim Hee Lee. “DReg-NeRF: Deep Registration for Neural Radiance Fields”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2023, pp. 22646–22656. DOI: [10.1109/iccv51070.2023.02075](https://doi.org/10.1109/iccv51070.2023.02075).
- [89] Danny Driess et al. *PaLM-E: An Embodied Multimodal Language Model*. 2023. DOI: [10.48550/arxiv.2303.03378](https://doi.org/10.48550/arxiv.2303.03378).

- [90] Nouha Dziri et al. *Faith and Fate: Limits of Transformers on Compositionality*. 2023. DOI: [10.48550/arxiv.2305.18654](https://doi.org/10.48550/arxiv.2305.18654).
- [91] Weixi Feng et al. *LayoutGPT: Compositional Visual Planning and Generation with Large Language Models*. 2023. DOI: [10.48550/arxiv.2305.15393](https://doi.org/10.48550/arxiv.2305.15393).
- [92] Samir Yitzhak Gadre et al. “CoWs on Pasture: Baselines and Benchmarks for Language-Driven Zero-Shot Object Navigation”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 23171–23181. DOI: [10.1109/cvpr52729.2023.02219](https://doi.org/10.1109/cvpr52729.2023.02219).
- [93] Haoran Geng et al. “GAPartNet: Cross-Category Domain-Generalizable Object Perception and Manipulation via Generalizable and Actionable Parts”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 7081–7091. DOI: [10.1109/cvpr52729.2023.00684](https://doi.org/10.1109/cvpr52729.2023.00684).
- [94] Haoran Geng et al. “PartManip: Learning Cross-Category Generalizable Part Manipulation Policy from Point Cloud Observations”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 2978–2988. DOI: [10.1109/cvpr52729.2023.00291](https://doi.org/10.1109/cvpr52729.2023.00291).
- [95] Lily Goli et al. “nerf2nerf: Pairwise Registration of Neural Radiance Fields”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023. DOI: [10.1109/icra48891.2023.10160794](https://doi.org/10.1109/icra48891.2023.10160794).
- [96] Ori Gordon, Omri Avrahami, and Dani Lischinski. “Blended-NeRF: Zero-Shot Object Generation and Blending in Existing Neural Radiance Fields”. In: *IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. IEEE, Oct. 2023, pp. 2933–2943. DOI: [10.1109/iccvw60793.2023.00316](https://doi.org/10.1109/iccvw60793.2023.00316).
- [97] Lin Guan et al. *Leveraging Pre-trained Large Language Models to Construct and Utilize World Models for Model-based Task Planning*. 2023. DOI: [10.48550/arxiv.2305.14909](https://doi.org/10.48550/arxiv.2305.14909).

- [98] Nick Heppert et al. “CARTO: Category and Joint Agnostic Reconstruction of ARTiculated Objects”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 21201–21210. DOI: [10.1109/cvpr52729.2023.02031](https://doi.org/10.1109/cvpr52729.2023.02031).
- [99] Cheng-Chun Hsu, Zhenyu Jiang, and Yuke Zhu. “Ditto in the House: Building Articulation Models of Indoor Scenes through Interactive Perception”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023, pp. 3933–3939. DOI: [10.1109/icra48891.2023.10161431](https://doi.org/10.1109/icra48891.2023.10161431).
- [100] Joy Hsu, Jiayuan Mao, and Jiajun Wu. “NS3D: Neuro-Symbolic Grounding of 3D Objects and Relations”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 2614–2623. DOI: [10.1109/cvpr52729.2023.00257](https://doi.org/10.1109/cvpr52729.2023.00257).
- [101] Chenguang Huang et al. “Visual Language Maps for Robot Navigation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023, pp. 10608–10615. DOI: [10.1109/icra48891.2023.10160969](https://doi.org/10.1109/icra48891.2023.10160969).
- [102] Wenlong Huang et al. “VoxPoser: Composable 3D Value Maps for Robotic Manipulation with Language Models”. In: *Conference on Robot Learning (CoRL)*. 2023. DOI: [10.48550/arxiv.2307.05973](https://doi.org/10.48550/arxiv.2307.05973).
- [103] Han Jiang et al. *Registering Neural Radiance Fields as 3D Density Images*. 2023. DOI: [10.48550/arxiv.2305.12843](https://doi.org/10.48550/arxiv.2305.12843).
- [104] Bernhard Kerbl et al. “3D Gaussian Splatting for Real-Time Radiance Field Rendering”. In: *ACM Transactions on Graphics (TOG)* 42.4 (July 2023), pp. 1–14. ISSN: 1557-7368. DOI: [10.1145/3592433](https://doi.org/10.1145/3592433).
- [105] Jacky Liang et al. “Code as Policies: Language Model Programs for Embodied Control”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023. DOI: [10.1109/icra48891.2023.10160591](https://doi.org/10.1109/icra48891.2023.10160591).
- [106] Yunzhi Lin et al. “Parallel Inversion of Neural Radiance Fields for Robust Pose Estimation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023, pp. 9377–9384. DOI: [10.1109/icra48891.2023.10161117](https://doi.org/10.1109/icra48891.2023.10161117).
- [107] Bo Liu et al. *LLM+P: Empowering Large Language Models with Optimal Planning Proficiency*. 2023. DOI: [10.48550/arxiv.2304.11477](https://doi.org/10.48550/arxiv.2304.11477).



- [108] Jiayi Liu, Ali Mahdavi-Amiri, and Manolis Savva. “PARIS: Part-level Reconstruction and Motion Analysis for Articulated Objects”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2023, pp. 352–363. DOI: [10.1109/iccv51070.2023.00039](https://doi.org/10.1109/iccv51070.2023.00039).
- [109] Shaowei Liu, Saurabh Gupta, and Shenlong Wang. “Building Rearticulable Models for Arbitrary 3D Objects from 4D Point Clouds”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 21138–21147. DOI: [10.1109/cvpr52729.2023.02025](https://doi.org/10.1109/cvpr52729.2023.02025).
- [110] Yecheng Jason Ma et al. *Eureka: Human-Level Reward Design via Coding Large Language Models*. 2023. DOI: [10.48550/arxiv.2310.12931](https://doi.org/10.48550/arxiv.2310.12931).
- [111] Dominic Maggio et al. “Loc-NeRF: Monte Carlo Localization using Neural Radiance Fields”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023, pp. 4018–4025. DOI: [10.1109/icra48891.2023.10160782](https://doi.org/10.1109/icra48891.2023.10160782).
- [112] OpenAI et al. *GPT-4 Technical Report*. 2023. DOI: [10.48550/arxiv.2303.08774](https://doi.org/10.48550/arxiv.2303.08774).
- [113] Shishir G. Patil et al. *Gorilla: Large Language Model Connected with Massive APIs*. 2023. DOI: [10.48550/arxiv.2305.15334](https://doi.org/10.48550/arxiv.2305.15334).
- [114] Allen Z. Ren et al. “Leveraging Language for Accelerated Learning of Tool Manipulation”. In: *Conference on Robot Learning (CoRL)*. 2023. DOI: [10.48550/arxiv.2206.13074](https://doi.org/10.48550/arxiv.2206.13074).
- [115] Timo Schick et al. *Toolformer: Language Models Can Teach Themselves to Use Tools*. 2023. DOI: [10.48550/arxiv.2302.04761](https://doi.org/10.48550/arxiv.2302.04761).
- [116] Jonas Schult et al. “Mask3D: Mask Transformer for 3D Semantic Instance Segmentation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2023, pp. 8216–8223. DOI: [10.1109/icra48891.2023.10160590](https://doi.org/10.1109/icra48891.2023.10160590).
- [117] Dhruv Shah et al. “LM-Nav: Robotic Navigation with Large Pre-Trained Models of Language, Vision, and Action”. In: *Conference on Robot Learning (CoRL)*. 2023. DOI: [10.48550/arxiv.2207.04429](https://doi.org/10.48550/arxiv.2207.04429).



- [118] Dídac Surís, Sachit Menon, and Carl Vondrick. “ViperGPT: Visual Inference via Python Execution for Reasoning”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2023, pp. 11854–11864. DOI: [10.1109/iccv51070.2023.01092](https://doi.org/10.1109/iccv51070.2023.01092).
- [119] Matthew Tancik et al. “Nerfstudio: A Modular Framework for Neural Radiance Field Development”. In: *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. SIGGRAPH ’23. ACM, July 2023, pp. 1–12. DOI: [10.1145/3588432.3591516](https://doi.org/10.1145/3588432.3591516).
- [120] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. DOI: [10.48550/arxiv.2307.09288](https://doi.org/10.48550/arxiv.2307.09288).
- [121] Karthik Valmeekam et al. *On the Planning Abilities of Large Language Models (A Critical Investigation with a Proposed Benchmark)*. 2023. DOI: [10.48550/arxiv.2302.06706](https://doi.org/10.48550/arxiv.2302.06706).
- [122] Renhao Wang et al. “Programmatically Grounded, Compositionally Generalizable Robotic Manipulation”. In: *International Conference on Learning Representations (ICLR)*. 2023. DOI: [10.48550/arxiv.2304.13826](https://doi.org/10.48550/arxiv.2304.13826).
- [123] Lionel Wong et al. *From Word Models to World Models: Translating from Natural Language to the Probabilistic Language of Thought*. 2023. DOI: [10.48550/arxiv.2306.12672](https://doi.org/10.48550/arxiv.2306.12672).
- [124] Jiannan Xiang et al. *Language Models Meet World Models: Embodied Experiences Enhance Language Models*. 2023. DOI: [10.48550/arxiv.2305.10626](https://doi.org/10.48550/arxiv.2305.10626).
- [125] Shunyu Yao et al. *Tree of Thoughts: Deliberate Problem Solving with Large Language Models*. 2023. DOI: [10.48550/arxiv.2305.10601](https://doi.org/10.48550/arxiv.2305.10601).
- [126] Bangguo Yu et al. *Co-NavGPT: Multi-Robot Cooperative Visual Semantic Navigation Using Vision Language Models*. 2023. DOI: [10.48550/arxiv.2310.07937](https://doi.org/10.48550/arxiv.2310.07937).
- [127] Wenhao Yu et al. “Language to Rewards for Robotic Skill Synthesis”. In: *Conference on Robot Learning (CoRL)*. 2023. DOI: [10.48550/arxiv.2306.08647](https://doi.org/10.48550/arxiv.2306.08647).

- [128] Xiaoshuai Zhang et al. “Nerflets: Local Radiance Fields for Efficient Structure-Aware 3D Scene Representation from 2D Supervision”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2023, pp. 8274–8284. DOI: [10.1109/cvpr52729.2023.00800](https://doi.org/10.1109/cvpr52729.2023.00800).
- [129] Ziyu Zhu et al. “3D-VisTA: Pre-trained Transformer for 3D Vision and Text Alignment”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2023, pp. 2899–2909. DOI: [10.1109/iccv51070.2023.00272](https://doi.org/10.1109/iccv51070.2023.00272).
- [130] Maciej Besta et al. “Graph of Thoughts: Solving Elaborate Problems with Large Language Models”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 38.16 (Mar. 2024), pp. 17682–17690. ISSN: 2159-5399. DOI: [10.1609/aaai.v38i16.29720](https://doi.org/10.1609/aaai.v38i16.29720).
- [131] Kevin Black et al.  $\pi_0$ : A Vision-Language-Action Flow Model for General Robot Control. 2024. DOI: [10.48550/arxiv.2410.24164](https://doi.org/10.48550/arxiv.2410.24164).
- [132] Jiahao Chang et al. “GaussReg: Fast 3D Registration with Gaussian Splatting”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, Nov. 2024, pp. 407–423. ISBN: 9783031726330. DOI: [10.1007/978-3-031-72633-0\\_23](https://doi.org/10.1007/978-3-031-72633-0_23).
- [133] Timothy Chen, Preston Culbertson, and Mac Schwager. “CATNIPS: Collision Avoidance through Neural Implicit Probabilistic Scenes”. In: *IEEE Transactions on Robotics (T-RO)* 40 (2024), pp. 2712–2728. ISSN: 1941-0468. DOI: [10.1109/tro.2024.3386394](https://doi.org/10.1109/tro.2024.3386394).
- [134] Yilun Chen et al. *Grounded 3D-LLM with Referent Tokens*. 2024. DOI: [10.48550/arxiv.2405.10370](https://doi.org/10.48550/arxiv.2405.10370).
- [135] An-Chieh Cheng et al. “SpatialRGPT: Grounded Spatial Reasoning in Vision Language Models”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2024. DOI: [10.48550/arxiv.2406.01584](https://doi.org/10.48550/arxiv.2406.01584).
- [136] Alexandros Delitzas et al. “SceneFun3D: Fine-Grained Functionality and Affordance Understanding in 3D Scenes”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, pp. 14531–14542. DOI: [10.1109/cvpr52733.2024.01377](https://doi.org/10.1109/cvpr52733.2024.01377).

- [137] Jiading Fang et al. “NeRFuser: Scalable Scene Representation by NeRF Registration and Blending”. In: *International Symposium of Robotics Research (ISRR)*. Springer International Publishing, Nov. 2024. DOI: [10.48550/arxiv.2305.13307](https://doi.org/10.48550/arxiv.2305.13307).
- [138] Jiading Fang et al. “Transcrib3D: 3D Referring Expression Resolution through Large Language Models”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2024, pp. 9737–9744. DOI: [10.1109/iros58592.2024.10802793](https://doi.org/10.1109/iros58592.2024.10802793).
- [139] Roya Firoozi et al. “Foundation models in robotics: Applications, challenges, and the future”. In: *International Journal of Robotics Research (IJRR)* 44.5 (Sept. 2024), pp. 701–739. ISSN: 1741-3176. DOI: [10.1177/02783649241281508](https://doi.org/10.1177/02783649241281508).
- [140] Dibya Ghosh et al. “Octo: An Open-Source Generalist Robot Policy”. In: *Robotics: Science and Systems (RSS)*. RSS2024. Robotics: Science and Systems Foundation, July 2024. DOI: [10.15607/rss.2024.xx.090](https://doi.org/10.15607/rss.2024.xx.090).
- [141] Antoine Guédon and Vincent Lepetit. “SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, pp. 5354–5363. DOI: [10.1109/cvpr52733.2024.00512](https://doi.org/10.1109/cvpr52733.2024.00512).
- [142] Haifeng Huang et al. “Chat-Scene: Bridging 3D Scene and Large Language Models with Object Identifiers”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. 2024. DOI: [10.48550/arxiv.2312.08168](https://doi.org/10.48550/arxiv.2312.08168).
- [143] Muhammad Zubair Irshad et al. *Neural Fields in Robotics: A Survey*. 2024. DOI: [10.48550/arxiv.2410.20220](https://doi.org/10.48550/arxiv.2410.20220).
- [144] Kento Kawaharazuka et al. “Real-world robot applications of foundation models: a review”. In: *Advanced Robotics* 38.18 (Sept. 2024), pp. 1232–1254. ISSN: 1568-5535. DOI: [10.1080/01691864.2024.2408593](https://doi.org/10.1080/01691864.2024.2408593).
- [145] Bernhard Kerbl et al. “A Hierarchical 3D Gaussian Representation for Real-Time Rendering of Very Large Datasets”. In: *ACM Transactions on Graphics (TOG)* 43.4 (July 2024), pp. 1–15. ISSN: 1557-7368. DOI: [10.1145/3658160](https://doi.org/10.1145/3658160).

- [146] Justin Kerr et al. “Robot See Robot Do: Imitating Articulated Object Manipulation with Monocular 4D Reconstruction”. In: *Conference on Robot Learning (CoRL)*. Vol. 270. Proceedings of Machine Learning Research. PMLR, 2024, pp. 587–603. DOI: [10.48550/arxiv.2409.18121](https://doi.org/10.48550/arxiv.2409.18121). eprint: [2409.18121](https://arxiv.org/abs/2409.18121).
- [147] Moo Jin Kim et al. “OpenVLA: An Open-Source Vision-Language-Action Model”. In: *Conference on Robot Learning (CoRL)*. 2024. DOI: [10.48550/arxiv.2406.09246](https://doi.org/10.48550/arxiv.2406.09246).
- [148] Jiaqi Lin et al. “VastGaussian: Vast 3D Gaussians for Large Scene Reconstruction”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, pp. 5166–5175. DOI: [10.1109/cvpr52733.2024.00494](https://doi.org/10.1109/cvpr52733.2024.00494).
- [149] Nelson F. Liu et al. “Lost in the Middle: How Language Models Use Long Contexts”. In: *Transactions of the Association for Computational Linguistics (TACL)* 12 (2024), pp. 157–173. ISSN: 2307-387X. DOI: [10.1162/tac1\\_a\\_00638](https://doi.org/10.1162/tac1_a_00638).
- [150] Yang Liu et al. “CityGaussian: Real-Time High-Quality Large-Scale Scene Rendering with Gaussians”. In: *European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland, Oct. 2024, pp. 265–282. ISBN: 9783031726408. DOI: [10.1007/978-3-031-72640-8\\_15](https://doi.org/10.1007/978-3-031-72640-8_15).
- [151] Tao Lu et al. “Scaffold-GS: Structured 3D Gaussians for View-Adaptive Rendering”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, pp. 20654–20664. DOI: [10.1109/cvpr52733.2024.01952](https://doi.org/10.1109/cvpr52733.2024.01952).
- [152] Abby O’Neill et al. “Open X-Embodiment: Robotic Learning Datasets and RT-X Models”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2024, pp. 6892–6903. DOI: [10.1109/icra57147.2024.10611477](https://doi.org/10.1109/icra57147.2024.10611477).
- [153] Nikolai Rozanov and Marek Rei. *StateAct: Enhancing LLM Base Agents via Self-prompting and State-tracking*. 2024. DOI: [10.48550/arxiv.2410.02810](https://doi.org/10.48550/arxiv.2410.02810).
- [154] Noah Shinn et al. “Reflexion: Language Agents with Verbal Reinforcement Learning”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 36 (2024). DOI: [10.48550/arxiv.2303.11366](https://doi.org/10.48550/arxiv.2303.11366).

- [155] Tom Silver et al. “Generalized Planning in PDDL Domains with Pre-trained Large Language Models”. In: *AAAI Conference on Artificial Intelligence (AAAI)* 38.18 (Mar. 2024), pp. 20256–20264. ISSN: 2159-5399. DOI: [10.1609/aaai.v38i18.30006](https://doi.org/10.1609/aaai.v38i18.30006).
- [156] Chaoyue Song et al. “REACTO: Reconstructing Articulated Objects from a Single Video”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, pp. 5384–5395. DOI: [10.1109/cvpr52733.2024.00515](https://doi.org/10.1109/cvpr52733.2024.00515).
- [157] Yijia Weng et al. “Neural Implicit Representation for Building Digital Twins of Unknown Articulated Objects”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2024, pp. 3141–3150. DOI: [10.1109/cvpr52733.2024.00303](https://doi.org/10.1109/cvpr52733.2024.00303).
- [158] Tianbao Xie et al. “Text2Reward: Reward Shaping with Language Models for Reinforcement Learning”. In: *International Conference on Learning Representations (ICLR)*. 2024. DOI: [10.48550/arxiv.2309.11489](https://doi.org/10.48550/arxiv.2309.11489).
- [159] Jianing Yang et al. *3D-GRAND: A Million-Scale Dataset for 3D-LLMs with Better Grounding and Less Hallucination*. 2024. DOI: [10.48550/arxiv.2406.05132](https://doi.org/10.48550/arxiv.2406.05132).
- [160] Jianing Yang et al. “LLM-Grounder: Open-Vocabulary 3D Visual Grounding with Large Language Model as an Agent”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2024, pp. 7694–7701. DOI: [10.1109/icra57147.2024.10610443](https://doi.org/10.1109/icra57147.2024.10610443).
- [161] Vickie Ye et al. *gsplat: An Open-Source Library for Gaussian Splatting*. 2024. DOI: [10.48550/arxiv.2409.06765](https://doi.org/10.48550/arxiv.2409.06765).
- [162] Takuma Yoneda et al. “Statler: State-Maintaining Language Models for Embodied Reasoning”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2024, pp. 15083–15091. DOI: [10.1109/icra57147.2024.10610634](https://doi.org/10.1109/icra57147.2024.10610634).
- [163] Sergey Zakharov et al. “ReFiNe: Recursive Field Networks for Cross-Modal Multi-Scene Representation”. In: *ACM Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)*. SIGGRAPH ’24. ACM, July 2024, pp. 1–9. DOI: [10.1145/3641519.3657526](https://doi.org/10.1145/3641519.3657526).

- [164] Chenming Zhu et al. *LLaVA-3D: A Simple yet Effective Pathway to Empowering LMMs with 3D-awareness*. 2024. DOI: [10.48550/arxiv.2409.18125](https://doi.org/10.48550/arxiv.2409.18125).
- [165] Zihan Zhu et al. “NICER-SLAM: Neural Implicit Scene Encoding for RGB SLAM”. In: *IEEE International Conference on 3D Vision (3DV)*. IEEE, Mar. 2024, pp. 42–52. DOI: [10.1109/3dv62453.2024.00096](https://doi.org/10.1109/3dv62453.2024.00096).
- [166] Zixuan Chen et al. *RoboHorizon: An LLM-Assisted Multi-View World Model for Long-Horizon Robotic Manipulation*. 2025. DOI: [10.48550/arxiv.2501.06605](https://doi.org/10.48550/arxiv.2501.06605).
- [167] DeepSeek-AI et al. *DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning*. 2025. DOI: [10.48550/arxiv.2501.12948](https://doi.org/10.48550/arxiv.2501.12948).
- [168] Jianning Deng, Kartic Subr, and Hakan Bilen. “Articulate your NeRF: Unsupervised articulated object modeling via conditional view synthesis”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 37. 2025, pp. 119717–119741. DOI: [10.48550/arxiv.2406.16623](https://doi.org/10.48550/arxiv.2406.16623). eprint: [2406.16623](https://arxiv.org/abs/2406.16623).
- [169] Junfu Guo et al. “ArticulatedGS: Self-supervised Digital Twin Modeling of Articulated Objects using 3D Gaussian Splatting”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2025, pp. 27144–27153. DOI: [10.1109/cvpr52734.2025.02528](https://doi.org/10.1109/cvpr52734.2025.02528).
- [170] Long Le et al. “Articulate-Anything: Automatic Modeling of Articulated Objects via a Vision-Language Foundation Model”. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2025. DOI: [10.48550/arxiv.2410.13882](https://doi.org/10.48550/arxiv.2410.13882).
- [171] Shengjie Lin et al. “SplArt: Articulation Estimation and Part-Level Reconstruction with 3D Gaussian Splatting”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, Oct. 2025. DOI: [10.48550/arxiv.2506.03594](https://doi.org/10.48550/arxiv.2506.03594).
- [172] Yu Liu et al. “ArtGS: Building Interactable Replicas of Complex Articulated Objects via Gaussian Splatting”. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2025. DOI: [10.48550/arxiv.2502.19459](https://doi.org/10.48550/arxiv.2502.19459).

- [173] Zhao Mandi et al. “Real2code: Reconstruct Articulated Objects Via Code Generation”. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2025. DOI: [10.48550/arxiv.2406.08474](https://doi.org/10.48550/arxiv.2406.08474).
- [174] Michael Niemeyer et al. “RadSplat: Radiance Field-Informed Gaussian Splatting for Robust Real- Time Rendering with 900+ FPS”. In: *IEEE International Conference on 3D Vision (3DV)*. IEEE, Mar. 2025, pp. 134–144. DOI: [10.1109/3dv66043.2025.00018](https://doi.org/10.1109/3dv66043.2025.00018).
- [175] Nikhila Ravi et al. “SAM 2: Segment Anything in Images and Videos”. In: *International Conference on Learning Representations (ICLR)*. OpenReview.net, 2025. DOI: [10.48550/arxiv.2408.00714](https://doi.org/10.48550/arxiv.2408.00714).
- [176] Zhenyu Tang et al. *NeuralGS: Bridging Neural Fields and 3D Gaussian Splatting for Compact 3D Representations*. 2025. DOI: [10.48550/arxiv.2503.23162](https://doi.org/10.48550/arxiv.2503.23162).
- [177] Abdelrhman Werby et al. *Articulated Object Estimation in the Wild*. 2025. DOI: [10.48550/arxiv.2509.01708](https://doi.org/10.48550/arxiv.2509.01708).
- [178] Di Wu et al. *REArtGS: Reconstructing and Generating Articulated Objects via 3D Gaussian Splatting with Geometric and Motion Constraints*. 2025. DOI: [10.48550/arxiv.2503.06677](https://doi.org/10.48550/arxiv.2503.06677).
- [179] Jiawei Yang et al. *STORM: Spatio-Temporal Reconstruction Model for Large-Scale Outdoor Scenes*. 2025. DOI: [10.48550/arxiv.2501.00602](https://doi.org/10.48550/arxiv.2501.00602).

# Appendix A

## Supplementary Material for SPLART

### A.1 Additional Qualitative Results

We present additional qualitative results in Figure A.2, showing a side-by-side comparison with the PARIS [108] baseline on part-level reconstruction and articulation estimation. Figure A.1 provides further examples of novel view synthesis for interpolated articulation states, highlighting SPLART’s ability to generate smooth and photorealistic transitions.

### A.2 Per-Scene Quantitative Results

For a detailed, per-scene breakdown of our quantitative evaluation, we refer the reader to the following tables.

- Table A.1: Articulation estimation results on the PARIS-PMS dataset.
- Table A.2: Articulation estimation results for revolute objects in the SPLART-PMS dataset.
- Table A.3: Articulation estimation results for prismatic objects in the SPLART-PMS dataset.
- Table A.4: Novel view synthesis results on the PARIS-PMS dataset, including PSNR, SSIM [2], LPIPS [14], and per-category IoU metrics.



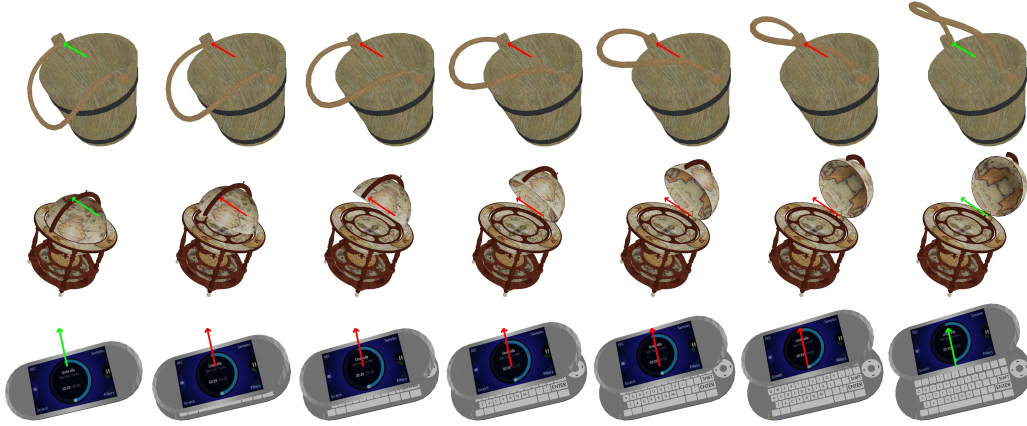


Figure A.1: Additional visualizations of state interpolation for three objects along with their estimated articulation models. The left- and right-most images correspond to the ground truth of the two end articulation states, annotated with the ground-truth articulation using **green arrows**. The intermediate images are rendered results obtained by interpolating the articulation state, annotated with both the ground-truth articulation (**green arrows**) and the estimated articulation (**red arrows**). Due to the high accuracy of the articulation estimation, the ground-truth **green arrows** are hidden by the estimated **red arrows** in the interpolated images.

- Table A.5: Novel view and articulation synthesis results for revolute objects in the SPLART-PMS dataset.
- Table A.6: Novel view and articulation synthesis results for prismatic objects in the SPLART-PMS dataset.

| Type      | Scene     | Method           | Success<br>rate $\uparrow$ | $\text{err}_a$<br>( $\times 10^{-2}$ DEG) $\downarrow$ |                 | $\text{err}_p$<br>( $\times 10^{-3}$ ) $\downarrow$ |  | $\text{err}_r$<br>( $\times 10^{-2}$ DEG) $\downarrow$ |  | $\text{err}_t$<br>( $\times 10^{-3}$ ) $\downarrow$ |  | $\text{CD}_s$<br>( $\times 10^{-3}$ ) $\downarrow$ |                    | $\text{CD}_m$<br>( $\times 10^{-3}$ ) $\downarrow$ |  | $\text{CD}_w$<br>( $\times 10^{-3}$ ) $\downarrow$ |  |
|-----------|-----------|------------------|----------------------------|--------------------------------------------------------|-----------------|-----------------------------------------------------|--|--------------------------------------------------------|--|-----------------------------------------------------|--|----------------------------------------------------|--------------------|----------------------------------------------------|--|----------------------------------------------------|--|
| Revolute  | USB       | PARIS            | 0/10                       | F                                                      |                 | F                                                   |  | F                                                      |  | N/A                                                 |  | F                                                  |                    | F                                                  |  | F                                                  |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 9.30 $\pm$ 1.29                                        | 0.39 $\pm$ 0.31 | 14.81 $\pm$ 3.49                                    |  | 14.81 $\pm$ 3.49                                       |  | N/A                                                 |  | 2.62 $\pm$ 0.06                                    | 1.48 $\pm$ 0.03    | 1.36 $\pm$ 0.02                                    |  | 1.36 $\pm$ 0.02                                    |  |
|           |           | SPLART           | 10/10                      | 4.74 $\pm$ 0.32                                        | 0.26 $\pm$ 0.05 | 3.99 $\pm$ 0.26                                     |  | 3.99 $\pm$ 0.26                                        |  | N/A                                                 |  | 0.89 $\pm$ 0.08                                    | 0.83 $\pm$ 0.11    | 0.91 $\pm$ 0.03                                    |  | 0.91 $\pm$ 0.03                                    |  |
|           | foldchair | PARIS            | 3/10                       | 103.68 $\pm$ 15.95                                     | 4.69 $\pm$ 4.77 | 195.70 $\pm$ 67.55                                  |  | 195.70 $\pm$ 67.55                                     |  | N/A                                                 |  | 0.45 $\pm$ 0.09                                    | 45.34 $\pm$ 4.62   | 14.85 $\pm$ 1.00                                   |  | 14.85 $\pm$ 1.00                                   |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 3.34 $\pm$ 1.43                                        | 0.50 $\pm$ 0.31 | 9.72 $\pm$ 3.61                                     |  | 9.72 $\pm$ 3.61                                        |  | N/A                                                 |  | 0.18 $\pm$ 0.00                                    | 0.14 $\pm$ 0.00    | 0.26 $\pm$ 0.00                                    |  | 0.26 $\pm$ 0.00                                    |  |
|           |           | SPLART           | 10/10                      | 5.47 $\pm$ 0.22                                        | 0.12 $\pm$ 0.03 | 8.93 $\pm$ 0.34                                     |  | 8.93 $\pm$ 0.34                                        |  | N/A                                                 |  | 1.53 $\pm$ 0.68                                    | 0.33 $\pm$ 0.01    | 0.31 $\pm$ 0.00                                    |  | 0.31 $\pm$ 0.00                                    |  |
|           | fridge    | PARIS            | 9/10                       | 105.51 $\pm$ 50.87                                     | 6.04 $\pm$ 4.17 | 121.25 $\pm$ 45.78                                  |  | 121.25 $\pm$ 45.78                                     |  | N/A                                                 |  | 2.97 $\pm$ 0.19                                    | 39.11 $\pm$ 2.07   | 11.62 $\pm$ 0.54                                   |  | 11.62 $\pm$ 0.54                                   |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 6.67 $\pm$ 2.79                                        | 0.51 $\pm$ 0.32 | 12.07 $\pm$ 2.08                                    |  | 12.07 $\pm$ 2.08                                       |  | N/A                                                 |  | 0.63 $\pm$ 0.01                                    | 0.29 $\pm$ 0.01    | 0.70 $\pm$ 0.00                                    |  | 0.70 $\pm$ 0.00                                    |  |
|           |           | SPLART           | 10/10                      | 4.14 $\pm$ 0.56                                        | 0.06 $\pm$ 0.05 | 5.81 $\pm$ 0.34                                     |  | 5.81 $\pm$ 0.34                                        |  | N/A                                                 |  | 2.23 $\pm$ 0.05                                    | 1.21 $\pm$ 0.03    | 1.95 $\pm$ 0.04                                    |  | 1.95 $\pm$ 0.04                                    |  |
|           | laptop    | PARIS            | 10/10                      | 117.26 $\pm$ 94.05                                     | 6.89 $\pm$ 5.45 | 99.11 $\pm$ 51.26                                   |  | 99.11 $\pm$ 51.26                                      |  | N/A                                                 |  | 0.61 $\pm$ 0.33                                    | 32.43 $\pm$ 2.19   | 15.08 $\pm$ 1.41                                   |  | 15.08 $\pm$ 1.41                                   |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 6.65 $\pm$ 1.73                                        | 1.33 $\pm$ 0.52 | 11.81 $\pm$ 3.62                                    |  | 11.81 $\pm$ 3.62                                       |  | N/A                                                 |  | 0.31 $\pm$ 0.00                                    | 0.14 $\pm$ 0.00    | 0.34 $\pm$ 0.00                                    |  | 0.34 $\pm$ 0.00                                    |  |
|           |           | SPLART           | 10/10                      | 3.20 $\pm$ 0.55                                        | 0.59 $\pm$ 0.05 | 4.89 $\pm$ 0.28                                     |  | 4.89 $\pm$ 0.28                                        |  | N/A                                                 |  | 0.24 $\pm$ 0.00                                    | 0.34 $\pm$ 0.01    | 0.35 $\pm$ 0.01                                    |  | 0.35 $\pm$ 0.01                                    |  |
|           | oven      | PARIS            | 10/10                      | 161.16 $\pm$ 112.94                                    | 3.44 $\pm$ 3.81 | 95.15 $\pm$ 55.90                                   |  | 95.15 $\pm$ 55.90                                      |  | N/A                                                 |  | 9.89 $\pm$ 0.93                                    | 156.89 $\pm$ 43.24 | 10.90 $\pm$ 1.70                                   |  | 10.90 $\pm$ 1.70                                   |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 19.35 $\pm$ 4.10                                       | 1.57 $\pm$ 0.60 | 11.03 $\pm$ 2.19                                    |  | 11.03 $\pm$ 2.19                                       |  | N/A                                                 |  | 4.61 $\pm$ 0.07                                    | 0.44 $\pm$ 0.01    | 4.26 $\pm$ 0.06                                    |  | 4.26 $\pm$ 0.06                                    |  |
|           |           | SPLART           | 10/10                      | 1.45 $\pm$ 0.32                                        | 0.90 $\pm$ 0.06 | 2.95 $\pm$ 0.27                                     |  | 2.95 $\pm$ 0.27                                        |  | N/A                                                 |  | 7.10 $\pm$ 0.44                                    | 1.89 $\pm$ 3.52    | 6.02 $\pm$ 0.32                                    |  | 6.02 $\pm$ 0.32                                    |  |
|           | scissor   | PARIS            | 0/10                       | F                                                      |                 | F                                                   |  | F                                                      |  | N/A                                                 |  | F                                                  |                    | F                                                  |  | F                                                  |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 7.28 $\pm$ 5.05                                        | 4.84 $\pm$ 7.66 | 56.56 $\pm$ 95.23                                   |  | 56.56 $\pm$ 95.23                                      |  | N/A                                                 |  | 1.67 $\pm$ 3.41                                    | 5.06 $\pm$ 13.99   | 0.42 $\pm$ 0.00                                    |  | 0.42 $\pm$ 0.00                                    |  |
|           |           | SPLART           | 10/10                      | 2.24 $\pm$ 0.17                                        | 0.24 $\pm$ 0.06 | 2.18 $\pm$ 0.26                                     |  | 2.18 $\pm$ 0.26                                        |  | N/A                                                 |  | 0.37 $\pm$ 0.01                                    | 0.20 $\pm$ 0.01    | 0.25 $\pm$ 0.00                                    |  | 0.25 $\pm$ 0.00                                    |  |
|           | stapler   | PARIS            | 0/10                       | F                                                      |                 | F                                                   |  | F                                                      |  | N/A                                                 |  | F                                                  |                    | F                                                  |  | F                                                  |  |
|           |           | DTA <sup>†</sup> | 9/10                       | 7.06 $\pm$ 5.68                                        | 1.81 $\pm$ 1.16 | 11.76 $\pm$ 11.74                                   |  | 11.76 $\pm$ 11.74                                      |  | N/A                                                 |  | 2.93 $\pm$ 0.24                                    | 2.16 $\pm$ 0.89    | 2.03 $\pm$ 0.03                                    |  | 2.03 $\pm$ 0.03                                    |  |
|           |           | SPLART           | 10/10                      | 4.64 $\pm$ 0.43                                        | 0.75 $\pm$ 0.08 | 4.09 $\pm$ 0.28                                     |  | 4.09 $\pm$ 0.28                                        |  | N/A                                                 |  | 1.17 $\pm$ 0.02                                    | 2.12 $\pm$ 0.14    | 1.05 $\pm$ 0.02                                    |  | 1.05 $\pm$ 0.02                                    |  |
|           | washer    | PARIS            | 0/10                       | F                                                      |                 | F                                                   |  | F                                                      |  | N/A                                                 |  | F                                                  |                    | F                                                  |  | F                                                  |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 38.95 $\pm$ 11.24                                      | 3.91 $\pm$ 2.63 | 27.51 $\pm$ 8.30                                    |  | 27.51 $\pm$ 8.30                                       |  | N/A                                                 |  | 4.68 $\pm$ 0.11                                    | 0.40 $\pm$ 0.01    | 4.46 $\pm$ 0.12                                    |  | 4.46 $\pm$ 0.12                                    |  |
|           |           | SPLART           | 10/10                      | 3.76 $\pm$ 0.48                                        | 0.24 $\pm$ 0.15 | 5.95 $\pm$ 0.67                                     |  | 5.95 $\pm$ 0.67                                        |  | N/A                                                 |  | 19.14 $\pm$ 1.27                                   | 1.59 $\pm$ 2.40    | 17.92 $\pm$ 1.22                                   |  | 17.92 $\pm$ 1.22                                   |  |
|           | mean      | PARIS            | 40.0%                      | 121.90 $\pm$ 68.45                                     | 5.27 $\pm$ 4.55 | 127.80 $\pm$ 55.12                                  |  | 127.80 $\pm$ 55.12                                     |  | N/A                                                 |  | 3.48 $\pm$ 0.39                                    | 68.44 $\pm$ 13.03  | 13.11 $\pm$ 1.16                                   |  | 13.11 $\pm$ 1.16                                   |  |
|           |           | DTA <sup>†</sup> | 98.8%                      | 12.32 $\pm$ 4.16                                       | 1.86 $\pm$ 1.69 | 19.41 $\pm$ 16.28                                   |  | 19.41 $\pm$ 16.28                                      |  | N/A                                                 |  | 2.20 $\pm$ 0.49                                    | 1.27 $\pm$ 1.87    | 1.73 $\pm$ 0.03                                    |  | 1.73 $\pm$ 0.03                                    |  |
|           |           | SPLART           | 100.0%                     | 3.70 $\pm$ 0.38                                        | 0.40 $\pm$ 0.07 | 4.85 $\pm$ 0.33                                     |  | 4.85 $\pm$ 0.33                                        |  | N/A                                                 |  | 4.08 $\pm$ 0.32                                    | 1.06 $\pm$ 0.78    | 3.59 $\pm$ 0.21                                    |  | 3.59 $\pm$ 0.21                                    |  |
| Prismatic | blade     | PARIS            | 0/10                       | F                                                      |                 | N/A                                                 |  | N/A                                                    |  | F                                                   |  | F                                                  |                    | F                                                  |  | F                                                  |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 25.72 $\pm$ 4.58                                       | N/A             | N/A                                                 |  | N/A                                                    |  | 0.94 $\pm$ 0.14                                     |  | 0.49 $\pm$ 0.01                                    | 31.11 $\pm$ 0.55   | 0.37 $\pm$ 0.01                                    |  | 0.37 $\pm$ 0.01                                    |  |
|           |           | SPLART           | 9/10                       | 2.04 $\pm$ 0.61                                        | N/A             | N/A                                                 |  | N/A                                                    |  | 0.22 $\pm$ 0.04                                     |  | 0.44 $\pm$ 0.01                                    | 26.60 $\pm$ 0.85   | 0.44 $\pm$ 0.01                                    |  | 0.44 $\pm$ 0.01                                    |  |
|           | storage   | PARIS            | 10/10                      | 27.97 $\pm$ 13.09                                      | N/A             | N/A                                                 |  | N/A                                                    |  | 4.28 $\pm$ 3.02                                     |  | 9.21 $\pm$ 1.94                                    | 151.78 $\pm$ 35.00 | 7.99 $\pm$ 0.49                                    |  | 7.99 $\pm$ 0.49                                    |  |
|           |           | DTA <sup>†</sup> | 10/10                      | 6.79 $\pm$ 2.50                                        | N/A             | N/A                                                 |  | N/A                                                    |  | 1.37 $\pm$ 0.13                                     |  | 4.88 $\pm$ 0.07                                    | 0.36 $\pm$ 0.00    | 4.07 $\pm$ 0.06                                    |  | 4.07 $\pm$ 0.06                                    |  |
|           |           | SPLART           | 10/10                      | 2.69 $\pm$ 0.26                                        | N/A             | N/A                                                 |  | N/A                                                    |  | 0.45 $\pm$ 0.04                                     |  | 11.60 $\pm$ 0.38                                   | 6.94 $\pm$ 1.43    | 6.94 $\pm$ 0.27                                    |  | 6.94 $\pm$ 0.27                                    |  |
|           | mean      | PARIS            | 50.0%                      | 27.97 $\pm$ 13.09                                      | N/A             | N/A                                                 |  | N/A                                                    |  | 4.28 $\pm$ 3.02                                     |  | 9.21 $\pm$ 1.94                                    | 151.78 $\pm$ 35.00 | 7.99 $\pm$ 0.49                                    |  | 7.99 $\pm$ 0.49                                    |  |
|           |           | DTA <sup>†</sup> | 100.0%                     | 16.26 $\pm$ 3.54                                       | N/A             | N/A                                                 |  | N/A                                                    |  | 1.15 $\pm$ 0.14                                     |  | 2.69 $\pm$ 0.04                                    | 15.74 $\pm$ 0.28   | 2.22 $\pm$ 0.03                                    |  | 2.22 $\pm$ 0.03                                    |  |
|           |           | SPLART           | 95.0%                      | 2.36 $\pm$ 0.44                                        | N/A             | N/A                                                 |  | N/A                                                    |  | 0.33 $\pm$ 0.04                                     |  | 6.02 $\pm$ 0.20                                    | 16.77 $\pm$ 1.14   | 3.69 $\pm$ 0.14                                    |  | 3.69 $\pm$ 0.14                                    |  |
|           | Overall   | PARIS            | 42.0%                      | 103.12 $\pm$ 57.38                                     | 5.27 $\pm$ 4.55 | 127.80 $\pm$ 55.12                                  |  | 127.80 $\pm$ 55.12                                     |  | 4.28 $\pm$ 3.02                                     |  | 4.63 $\pm$ 0.70                                    | 85.11 $\pm$ 17.42  | 12.09 $\pm$ 1.03                                   |  | 12.09 $\pm$ 1.03                                   |  |
|           |           | DTA <sup>†</sup> | 99.0%                      | 13.11 $\pm$ 4.04                                       | 1.86 $\pm$ 1.69 | 19.41 $\pm$ 16.28                                   |  | 19.41 $\pm$ 16.28                                      |  | 1.15 $\pm$ 0.14                                     |  | 2.30 $\pm$ 0.40                                    | 4.16 $\pm$ 1.55    | 1.83 $\pm$ 0.03                                    |  | 1.83 $\pm$ 0.03                                    |  |
|           |           | SPLART           | 99.0%                      | 3.44 $\pm$ 0.39                                        | 0.40 $\pm$ 0.07 | 4.85 $\pm$ 0.33                                     |  | 4.85 $\pm$ 0.33                                        |  | 0.33 $\pm$ 0.04                                     |  | 4.47 $\pm$ 0.30                                    | 4.20 $\pm$ 0.85    | 3.61 $\pm$ 0.19                                    |  | 3.61 $\pm$ 0.19                                    |  |

Table A.1: PARIS-PMS articulation and mesh reconstruction metrics. <sup>†</sup>DTA requires ground-truth depth.

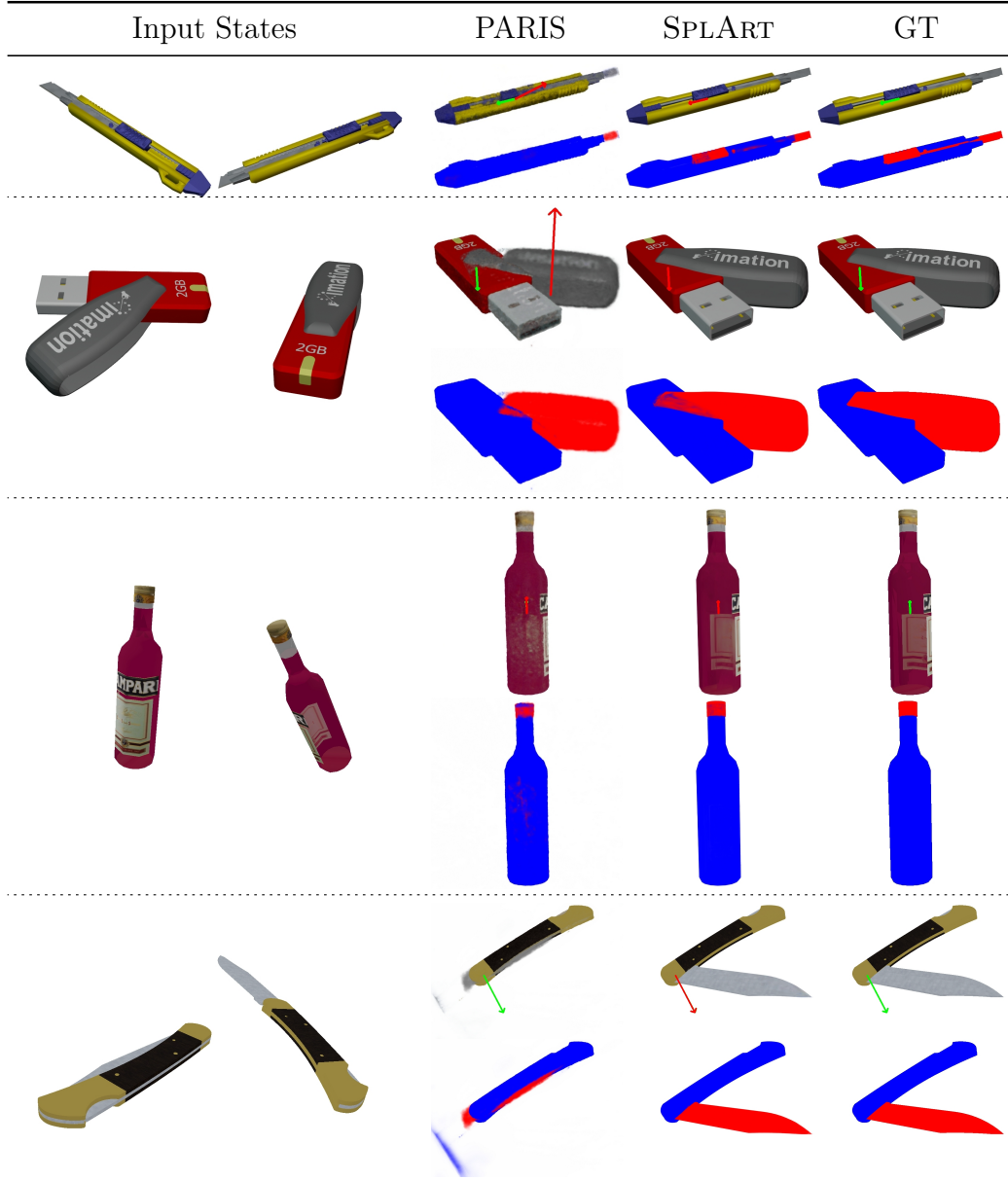


Figure A.2: Additional qualitative comparisons of part-level reconstruction and articulation estimation. For each object, the first row shows color renderings and the second row shows part segmentation maps. “GT” denotes ground truth.

| Scene                   | Method           | Success<br>rate $\uparrow$ | $\text{err}_a$<br>( $\times 10^{-2}$ DEG) $\downarrow$ | $\text{err}_p$<br>( $\times 10^{-3}$ ) $\downarrow$ | $\text{err}_r$<br>( $\times 10^{-2}$ DEG) $\downarrow$ |
|-------------------------|------------------|----------------------------|--------------------------------------------------------|-----------------------------------------------------|--------------------------------------------------------|
| 2230<br>Chair           | PARIS            | 1/10                       | 44.97 $\pm$ 0.00                                       | 4.41 $\pm$ 0.00                                     | 130.66 $\pm$ 0.00                                      |
|                         | DTA <sup>†</sup> | 9/10                       | 2.77 $\pm$ 0.79                                        | 0.18 $\pm$ 0.14                                     | 5.62 $\pm$ 2.35                                        |
|                         | SPLART           | 10/10                      | 0.33 $\pm$ 0.13                                        | 0.06 $\pm$ 0.03                                     | 0.84 $\pm$ 0.36                                        |
| 5477<br>Display         | PARIS            | 8/10                       | 86.76 $\pm$ 27.41                                      | 3.42 $\pm$ 3.21                                     | 133.12 $\pm$ 43.26                                     |
|                         | DTA <sup>†</sup> | 10/10                      | 1.38 $\pm$ 0.39                                        | 0.27 $\pm$ 0.06                                     | 2.29 $\pm$ 0.55                                        |
|                         | SPLART           | 10/10                      | 1.91 $\pm$ 0.43                                        | 0.49 $\pm$ 0.25                                     | 4.10 $\pm$ 0.75                                        |
| 7054<br>Clock           | PARIS            | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 1/10                       | 211.22 $\pm$ 0.00                                      | 1.06 $\pm$ 0.00                                     | 390.40 $\pm$ 0.00                                      |
| 11951<br>TrashCan       | PARIS            | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 10/10                      | 1.65 $\pm$ 0.71                                        | 21.55 $\pm$ 16.33                                   | 473.55 $\pm$ 352.80                                    |
| 100247<br>Box           | PARIS            | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 8/10                       | 0.79 $\pm$ 0.31                                        | 0.14 $\pm$ 0.10                                     | 1.99 $\pm$ 0.67                                        |
| 100460<br>Bucket        | PARIS            | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | DTA <sup>†</sup> | 10/10                      | 15.09 $\pm$ 6.89                                       | 0.61 $\pm$ 0.34                                     | 24.57 $\pm$ 10.06                                      |
|                         | SPLART           | 10/10                      | 0.50 $\pm$ 0.23                                        | 0.03 $\pm$ 0.04                                     | 0.79 $\pm$ 0.36                                        |
| 100756<br>Globe         | PARIS            | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 1/10                       | 0.72 $\pm$ 0.00                                        | 0.09 $\pm$ 0.00                                     | 3.31 $\pm$ 0.00                                        |
| 100794<br>Globe         | PARIS            | 1/10                       | 126.06 $\pm$ 0.00                                      | 14.17 $\pm$ 0.00                                    | 533.60 $\pm$ 0.00                                      |
|                         | DTA <sup>†</sup> | 10/10                      | 4.59 $\pm$ 3.84                                        | 2.67 $\pm$ 1.39                                     | 108.22 $\pm$ 10.35                                     |
|                         | SPLART           | 10/10                      | 1.22 $\pm$ 0.35                                        | 0.12 $\pm$ 0.10                                     | 2.09 $\pm$ 0.50                                        |
| 100882<br>Switch        | PARIS            | 2/10                       | 393.07 $\pm$ 86.79                                     | 15.78 $\pm$ 4.03                                    | 342.44 $\pm$ 171.13                                    |
|                         | DTA <sup>†</sup> | 1/10                       | 139.96 $\pm$ 0.00                                      | 2.67 $\pm$ 0.00                                     | 63.28 $\pm$ 0.00                                       |
|                         | SPLART           | 10/10                      | 19.37 $\pm$ 1.29                                       | 0.62 $\pm$ 0.23                                     | 11.20 $\pm$ 0.61                                       |
| 101542<br>Dispenser     | PARIS            | 7/10                       | 71.74 $\pm$ 49.72                                      | 5.44 $\pm$ 6.19                                     | 251.86 $\pm$ 267.19                                    |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 8/10                       | 0.75 $\pm$ 0.43                                        | 0.19 $\pm$ 0.06                                     | 11.48 $\pm$ 0.70                                       |
| 102400<br>Knife         | PARIS            | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 5/10                       | 9.01 $\pm$ 5.46                                        | 22.59 $\pm$ 11.58                                   | 37.95 $\pm$ 15.46                                      |
| 103031<br>CoffeeMachine | PARIS            | 3/10                       | 281.43 $\pm$ 90.89                                     | 12.95 $\pm$ 8.94                                    | 230.56 $\pm$ 65.56                                     |
|                         | DTA <sup>†</sup> | 0/10                       | F                                                      | F                                                   | F                                                      |
|                         | SPLART           | 10/10                      | 1.69 $\pm$ 0.40                                        | 0.22 $\pm$ 0.14                                     | 1.98 $\pm$ 0.58                                        |
| mean                    | PARIS            | 18.3%                      | 167.34 $\pm$ 42.47                                     | 9.36 $\pm$ 3.73                                     | 270.37 $\pm$ 91.19                                     |
|                         | DTA <sup>†</sup> | 33.3%                      | 32.76 $\pm$ 2.38                                       | 1.28 $\pm$ 0.39                                     | 40.80 $\pm$ 4.66                                       |
|                         | SPLART           | 77.5%                      | 20.76 $\pm$ 0.81                                       | 3.93 $\pm$ 2.41                                     | 78.31 $\pm$ 31.06                                      |

Table A.2: SPLART-PMS articulation metrics on revolute scenes. <sup>†</sup>DTA requires ground-truth depth.

| Scene               | Method           | Success<br>rate $\uparrow$ | $\text{err}_a$<br>( $\times 10^{-2}$ DEG) $\downarrow$ |        | $\text{err}_t$<br>( $\times 10^{-3}$ ) $\downarrow$ |       |
|---------------------|------------------|----------------------------|--------------------------------------------------------|--------|-----------------------------------------------------|-------|
| 3558<br>Bottle      | PARIS            | 5/10                       | 45.83 $\pm$                                            | 22.38  | 23.32 $\pm$                                         | 12.29 |
|                     | DTA <sup>†</sup> | 10/10                      | 200.52 $\pm$                                           | 10.32  | 4.60 $\pm$                                          | 0.27  |
|                     | SPLART           | 10/10                      | 3.82 $\pm$                                             | 0.52   | 0.10 $\pm$                                          | 0.02  |
| 12085<br>Dishwasher | PARIS            | 6/10                       | 15.60 $\pm$                                            | 7.00   | 4.67 $\pm$                                          | 1.10  |
|                     | DTA <sup>†</sup> | 10/10                      | 3.78 $\pm$                                             | 1.96   | 2.46 $\pm$                                          | 0.18  |
|                     | SPLART           | 10/10                      | 1.00 $\pm$                                             | 0.14   | 0.24 $\pm$                                          | 0.02  |
| 27189<br>Table      | PARIS            | 3/10                       | 23.69 $\pm$                                            | 10.43  | 21.18 $\pm$                                         | 3.03  |
|                     | DTA <sup>†</sup> | 10/10                      | 15.90 $\pm$                                            | 2.43   | 2.69 $\pm$                                          | 0.45  |
|                     | SPLART           | 10/10                      | 0.14 $\pm$                                             | 0.05   | 0.14 $\pm$                                          | 0.01  |
| 100248<br>Suitcase  | PARIS            | 0/10                       | F                                                      |        | F                                                   |       |
|                     | DTA <sup>†</sup> | 10/10                      | 192.55 $\pm$                                           | 30.46  | 10.57 $\pm$                                         | 1.60  |
|                     | SPLART           | 10/10                      | 1.22 $\pm$                                             | 0.58   | 0.10 $\pm$                                          | 0.04  |
| 101713<br>Pen       | PARIS            | 0/10                       | F                                                      |        | F                                                   |       |
|                     | DTA <sup>†</sup> | 10/10                      | 179.16 $\pm$                                           | 51.33  | 3.76 $\pm$                                          | 1.04  |
|                     | SPLART           | 10/10                      | 244.39 $\pm$                                           | 172.86 | 7.96 $\pm$                                          | 2.71  |
| 102016<br>USB       | PARIS            | 2/10                       | 17.52 $\pm$                                            | 10.01  | 9.53 $\pm$                                          | 1.87  |
|                     | DTA <sup>†</sup> | 10/10                      | 44.11 $\pm$                                            | 5.59   | 7.34 $\pm$                                          | 0.69  |
|                     | SPLART           | 10/10                      | 0.71 $\pm$                                             | 0.33   | 0.49 $\pm$                                          | 0.09  |
| 102812<br>Switch    | PARIS            | 8/10                       | 34.54 $\pm$                                            | 18.03  | 9.04 $\pm$                                          | 6.90  |
|                     | DTA <sup>†</sup> | 10/10                      | 190.87 $\pm$                                           | 8.20   | 38.68 $\pm$                                         | 1.85  |
|                     | SPLART           | 10/10                      | 1.55 $\pm$                                             | 0.80   | 0.11 $\pm$                                          | 0.05  |
| 103042<br>Window    | PARIS            | 5/10                       | 189.87 $\pm$                                           | 54.52  | 30.48 $\pm$                                         | 9.55  |
|                     | DTA <sup>†</sup> | 10/10                      | 92.93 $\pm$                                            | 1.64   | 11.92 $\pm$                                         | 0.23  |
|                     | SPLART           | 10/10                      | 1.55 $\pm$                                             | 0.24   | 0.76 $\pm$                                          | 0.18  |
| 103549<br>Toaster   | PARIS            | 0/10                       | F                                                      |        | F                                                   |       |
|                     | DTA <sup>†</sup> | 0/10                       | F                                                      |        | F                                                   |       |
|                     | SPLART           | 5/10                       | 2.63 $\pm$                                             | 0.74   | 0.30 $\pm$                                          | 0.07  |
| 103941<br>Phone     | PARIS            | 10/10                      | 19.39 $\pm$                                            | 10.07  | 6.51 $\pm$                                          | 3.81  |
|                     | DTA <sup>†</sup> | 10/10                      | 5.06 $\pm$                                             | 0.42   | 1.28 $\pm$                                          | 0.28  |
|                     | SPLART           | 10/10                      | 0.18 $\pm$                                             | 0.09   | 0.20 $\pm$                                          | 0.03  |
| mean                | PARIS            | 39.0%                      | 49.49 $\pm$                                            | 18.92  | 14.96 $\pm$                                         | 5.51  |
|                     | DTA <sup>†</sup> | 90.0%                      | 102.77 $\pm$                                           | 12.48  | 9.25 $\pm$                                          | 0.73  |
|                     | SPLART           | 95.0%                      | 25.72 $\pm$                                            | 17.63  | 1.04 $\pm$                                          | 0.32  |

Table A.3: SPLART-PMS articulation metrics on prismatic scenes. <sup>†</sup>DTA requires ground-truth depth.

| Type      | Scene            | Method           | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ | Depth MAE $\downarrow$ | IoU <sub>s</sub> $\uparrow$ | IoU <sub>m</sub> $\uparrow$ | IoU <sub>bg</sub> $\uparrow$ | mIoU $\uparrow$ |
|-----------|------------------|------------------|-----------------|-----------------|--------------------|------------------------|-----------------------------|-----------------------------|------------------------------|-----------------|
| Revolute  | USB              | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.022                  | 0.864                       | 0.892                       | 0.996                        | 0.917           |
|           |                  | SPLART           | 45.70           | 0.996           | 0.0029             | 0.026                  | 0.982                       | 0.979                       | 1.000                        | 0.987           |
|           | foldchair        | PARIS            | 33.48           | 0.936           | 0.0844             | 0.096                  | 0.961                       | 0.939                       | 0.994                        | 0.965           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.041                  | 0.915                       | 0.950                       | 0.988                        | 0.951           |
|           |                  | SPLART           | 43.98           | 0.992           | 0.0119             | 0.037                  | 0.981                       | 0.980                       | 0.999                        | 0.987           |
|           | fridge           | PARIS            | 32.22           | 0.967           | 0.0766             | 0.087                  | 0.973                       | 0.866                       | 0.997                        | 0.945           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.034                  | 0.957                       | 0.823                       | 0.992                        | 0.924           |
|           |                  | SPLART           | 41.40           | 0.996           | 0.0101             | 0.045                  | 0.986                       | 0.917                       | 0.999                        | 0.967           |
|           | laptop           | PARIS            | 31.64           | 0.964           | 0.0758             | 0.057                  | 0.907                       | 0.959                       | 0.997                        | 0.954           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.014                  | 0.938                       | 0.964                       | 0.997                        | 0.966           |
|           |                  | SPLART           | 40.90           | 0.995           | 0.0096             | 0.026                  | 0.931                       | 0.975                       | 0.999                        | 0.968           |
|           | oven             | PARIS            | 31.48           | 0.950           | 0.1048             | 0.132                  | 0.977                       | 0.891                       | 0.996                        | 0.955           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.038                  | 0.981                       | 0.946                       | 0.993                        | 0.973           |
|           |                  | SPLART           | 41.59           | 0.993           | 0.0148             | 0.082                  | 0.990                       | 0.935                       | 0.999                        | 0.974           |
|           | scissor          | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.051                  | 0.863                       | 0.876                       | 0.991                        | 0.910           |
|           |                  | SPLART           | 46.15           | 0.996           | 0.0030             | 0.044                  | 0.951                       | 0.954                       | 0.999                        | 0.968           |
|           | stapler          | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.035                  | 0.902                       | 0.894                       | 0.994                        | 0.930           |
|           |                  | SPLART           | 44.81           | 0.994           | 0.0021             | 0.034                  | 0.959                       | 0.940                       | 1.000                        | 0.966           |
|           | washer           | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.016                  | 0.991                       | 0.888                       | 0.998                        | 0.959           |
|           |                  | SPLART           | 43.75           | 0.996           | 0.0079             | 0.131                  | 0.995                       | 0.930                       | 0.999                        | 0.975           |
|           | mean             | PARIS            | 32.21           | 0.954           | 0.0854             | 0.093                  | 0.954                       | 0.914                       | 0.996                        | 0.955           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.031                  | 0.926                       | 0.904                       | 0.994                        | 0.941           |
|           |                  | SPLART           | 43.53           | 0.995           | 0.0078             | 0.053                  | 0.972                       | 0.951                       | 0.999                        | 0.974           |
| Prismatic | blade            | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.116                  | 0.728                       | 0.416                       | 0.998                        | 0.714           |
|           |                  | SPLART           | 46.15           | 0.998           | 0.0016             | 0.052                  | 0.900                       | 0.657                       | 1.000                        | 0.852           |
|           | storage          | PARIS            | 33.75           | 0.944           | 0.1228             | 0.108                  | 0.954                       | 0.722                       | 0.996                        | 0.891           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.017                  | 0.984                       | 0.944                       | 0.997                        | 0.975           |
|           |                  | SPLART           | 42.62           | 0.985           | 0.0481             | 0.037                  | 0.951                       | 0.850                       | 0.999                        | 0.933           |
|           | mean             | PARIS            | 33.75           | 0.944           | 0.1228             | 0.108                  | 0.954                       | 0.722                       | 0.996                        | 0.891           |
|           |                  | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.066                  | 0.856                       | 0.680                       | 0.997                        | 0.844           |
|           |                  | SPLART           | 44.38           | 0.991           | 0.0248             | 0.044                  | 0.925                       | 0.754                       | 0.999                        | 0.893           |
| Overall   | PARIS            | 32.52            | 0.952           | 0.0929          | 0.096              | 0.954                  | 0.875                       | 0.996                       | 0.942                        |                 |
|           | DTA <sup>†</sup> | N/A              | N/A             | N/A             | 0.038              | 0.912                  | 0.859                       | 0.994                       | 0.922                        |                 |
|           | SPLART           | 43.70            | 0.994           | 0.0112          | 0.052              | 0.963                  | 0.912                       | 0.999                       | 0.958                        |                 |

Table A.4: PARIS-PMS novel view synthesis metrics. <sup>†</sup>DTA requires ground-truth depth.

| Scene                   | Method           | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ | Depth MAE $\downarrow$ | IoU <sub>s</sub> $\uparrow$ | IoU <sub>m</sub> $\uparrow$ | IoU <sub>bg</sub> $\uparrow$ | mIoU $\uparrow$ |
|-------------------------|------------------|-----------------|-----------------|--------------------|------------------------|-----------------------------|-----------------------------|------------------------------|-----------------|
| 2230<br>Chair           | PARIS            | 28.89           | 0.924           | 0.1004             | 0.231                  | 0.510                       | 0.737                       | 0.982                        | 0.743           |
|                         | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.098                  | 0.764                       | 0.857                       | 0.980                        | 0.867           |
|                         | SPLART           | 32.08           | 0.972           | 0.0392             | 0.027                  | 0.498                       | 0.791                       | 0.993                        | 0.761           |
| 5477<br>Display         | PARIS            | 34.34           | 0.943           | 0.0979             | 0.100                  | 0.933                       | 0.973                       | 0.997                        | 0.968           |
|                         | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.020                  | 0.783                       | 0.968                       | 0.996                        | 0.916           |
|                         | SPLART           | 38.33           | 0.970           | 0.0379             | 0.020                  | 0.804                       | 0.971                       | 0.999                        | 0.924           |
| 7054<br>Clock           | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 35.54           | 0.985           | 0.0187             | 0.027                  | 0.994                       | 0.811                       | 0.999                        | 0.935           |
| 11951<br>TrashCan       | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 30.01           | 0.952           | 0.0560             | 0.083                  | 0.920                       | 0.735                       | 0.980                        | 0.878           |
| 100247<br>Box           | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 36.96           | 0.968           | 0.0409             | 0.030                  | 0.989                       | 0.987                       | 0.999                        | 0.992           |
| 100460<br>Bucket        | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.081                  | 0.948                       | 0.455                       | 0.984                        | 0.795           |
|                         | SPLART           | 37.83           | 0.970           | 0.0397             | 0.016                  | 0.989                       | 0.832                       | 0.999                        | 0.940           |
| 100756<br>Globe         | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 36.30           | 0.990           | 0.0346             | 0.035                  | 0.577                       | 0.346                       | 0.997                        | 0.640           |
| 100794<br>Globe         | PARIS            | 29.45           | 0.915           | 0.0965             | 0.115                  | 0.974                       | 0.937                       | 0.996                        | 0.969           |
|                         | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.046                  | 0.961                       | 0.943                       | 0.994                        | 0.966           |
|                         | SPLART           | 35.47           | 0.980           | 0.0216             | 0.012                  | 0.989                       | 0.985                       | 0.999                        | 0.991           |
| 100882<br>Switch        | PARIS            | 40.23           | 0.991           | 0.0418             | 0.082                  | 0.989                       | 0.822                       | 0.999                        | 0.937           |
|                         | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.020                  | 0.976                       | 0.676                       | 0.998                        | 0.883           |
|                         | SPLART           | 44.04           | 0.997           | 0.0174             | 0.033                  | 0.990                       | 0.835                       | 0.999                        | 0.941           |
| 101542<br>Dispenser     | PARIS            | 31.32           | 0.966           | 0.0450             | 0.105                  | 0.883                       | 0.623                       | 0.998                        | 0.835           |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 36.53           | 0.992           | 0.0153             | 0.020                  | 0.869                       | 0.585                       | 0.999                        | 0.818           |
| 102400<br>Knife         | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 36.21           | 0.990           | 0.0215             | 0.041                  | 0.986                       | 0.874                       | 0.997                        | 0.952           |
| 103031<br>CoffeeMachine | PARIS            | 32.29           | 0.979           | 0.0659             | 0.116                  | 0.977                       | 0.825                       | 0.997                        | 0.933           |
|                         | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                         | SPLART           | 37.20           | 0.993           | 0.0284             | 0.034                  | 0.994                       | 0.975                       | 0.999                        | 0.990           |
| mean                    | PARIS            | 32.75           | 0.953           | 0.0746             | 0.125                  | 0.878                       | 0.820                       | 0.995                        | 0.897           |
|                         | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.053                  | 0.886                       | 0.780                       | 0.990                        | 0.886           |
|                         | SPLART           | 36.38           | 0.980           | 0.0309             | 0.032                  | 0.883                       | 0.811                       | 0.997                        | 0.897           |

Table A.5: SPLART-PMS novel view and articulation synthesis metrics on revolute scenes. <sup>†</sup>DTA requires ground-truth depth.

| Scene               | Method           | PSNR $\uparrow$ | SSIM $\uparrow$ | LPIPS $\downarrow$ | Depth MAE $\downarrow$ | IoU <sub>s</sub> $\uparrow$ | IoU <sub>m</sub> $\uparrow$ | IoU <sub>bg</sub> $\uparrow$ | mIoU $\uparrow$ |
|---------------------|------------------|-----------------|-----------------|--------------------|------------------------|-----------------------------|-----------------------------|------------------------------|-----------------|
| 3558<br>Bottle      | PARIS            | 35.75           | 0.981           | 0.0276             | 0.120                  | 0.976                       | 0.676                       | 0.999                        | 0.883           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.026                  | 0.929                       | 0.580                       | 0.999                        | 0.836           |
|                     | SPLART           | 42.05           | 0.996           | 0.0076             | 0.034                  | 0.994                       | 0.968                       | 1.000                        | 0.987           |
| 12085<br>Dishwasher | PARIS            | 30.81           | 0.954           | 0.0927             | 0.111                  | 0.964                       | 0.929                       | 0.994                        | 0.962           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.113                  | 0.873                       | 0.773                       | 0.982                        | 0.876           |
|                     | SPLART           | 35.70           | 0.988           | 0.0388             | 0.019                  | 0.985                       | 0.965                       | 0.998                        | 0.983           |
| 27189<br>Table      | PARIS            | 29.55           | 0.887           | 0.1618             | 0.131                  | 0.958                       | 0.866                       | 0.995                        | 0.940           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.044                  | 0.964                       | 0.889                       | 0.992                        | 0.948           |
|                     | SPLART           | 34.76           | 0.963           | 0.0560             | 0.035                  | 0.979                       | 0.949                       | 0.998                        | 0.975           |
| 100248<br>Suitcase  | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.025                  | 0.986                       | 0.574                       | 0.998                        | 0.853           |
|                     | SPLART           | 39.76           | 0.995           | 0.0104             | 0.038                  | 0.995                       | 0.934                       | 0.999                        | 0.976           |
| 101713<br>Pen       | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.037                  | 0.956                       | 0.319                       | 0.999                        | 0.758           |
|                     | SPLART           | 40.85           | 0.998           | 0.0135             | 0.028                  | 0.981                       | 0.789                       | 1.000                        | 0.923           |
| 102016<br>USB       | PARIS            | 31.45           | 0.970           | 0.1005             | 0.191                  | 0.731                       | 0.583                       | 0.994                        | 0.769           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.157                  | 0.710                       | 0.553                       | 0.979                        | 0.747           |
|                     | SPLART           | 35.08           | 0.987           | 0.0475             | 0.054                  | 0.659                       | 0.533                       | 0.998                        | 0.730           |
| 102812<br>Switch    | PARIS            | 35.23           | 0.984           | 0.0433             | 0.102                  | 0.987                       | 0.908                       | 0.998                        | 0.964           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.024                  | 0.970                       | 0.616                       | 0.996                        | 0.861           |
|                     | SPLART           | 38.69           | 0.995           | 0.0235             | 0.031                  | 0.995                       | 0.971                       | 0.999                        | 0.988           |
| 103042<br>Window    | PARIS            | 29.11           | 0.967           | 0.0808             | 0.127                  | 0.859                       | 0.631                       | 0.996                        | 0.829           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.088                  | 0.900                       | 0.744                       | 0.991                        | 0.878           |
|                     | SPLART           | 32.35           | 0.984           | 0.0499             | 0.039                  | 0.868                       | 0.695                       | 0.999                        | 0.854           |
| 103549<br>Toaster   | PARIS            | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                     | DTA <sup>†</sup> | F               | F               | F                  | F                      | F                           | F                           | F                            | F               |
|                     | SPLART           | 39.04           | 0.994           | 0.0195             | 0.036                  | 0.997                       | 0.949                       | 0.999                        | 0.982           |
| 103941<br>Phone     | PARIS            | 34.21           | 0.972           | 0.0592             | 0.089                  | 0.938                       | 0.956                       | 0.998                        | 0.964           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.027                  | 0.913                       | 0.952                       | 0.997                        | 0.954           |
|                     | SPLART           | 39.42           | 0.993           | 0.0219             | 0.020                  | 0.971                       | 0.980                       | 0.999                        | 0.984           |
| mean                | PARIS            | 32.30           | 0.959           | 0.0808             | 0.124                  | 0.916                       | 0.793                       | 0.996                        | 0.902           |
|                     | DTA <sup>†</sup> | N/A             | N/A             | N/A                | 0.060                  | 0.911                       | 0.667                       | 0.993                        | 0.857           |
|                     | SPLART           | 37.77           | 0.989           | 0.0288             | 0.033                  | 0.942                       | 0.873                       | 0.999                        | 0.938           |

Table A.6: SPLART-PMS novel view and articulation synthesis metrics on prismatic scenes. <sup>†</sup>DTA requires ground-truth depth.



# Appendix B

## Supplementary Material for Statler

### B.1 Example Evaluation Episodes

In this section, we show three example evaluation episodes (one per domain), to give a better idea of what Statler is tasked to handle at test time. The episodes are shown in Listings B.1 to B.3. The *temporal* queries that require temporal reasoning to answer correctly are highlighted in blue. Python snippet under [Gold Code] shows the “correct” code the model is expected to generate. We note that the model can also output comments, `noop` and `update_wm` functions (in the case of Statler) that are not included in the gold code here. The whole set of episodes (20 episodes per domain) can be found in our released code repository.

### B.2 Prompts

For each domain, we provide a tailored prompt that consists of an example sequence of user queries and the expected Python code. To provide a concrete idea of our prompt design, we show our prompt of Code-as-Policies and Statler for the disinfection domain in Listings B.4 and B.5.

Listing B.1: Example evaluation episode (Pick-and-Place domain). The queries in blue require temporal reasoning.

---

```

1 [Initial State]
2 # state = {
3 #   "objects": ["green block", "orange block", "white block", "cyan block", "golden
   bowl", "red bowl"],
4 #   "relations": [],
5 #   "green block": {},
6 #   "orange block": {},
7 #   "white block": {},
8 #   "cyan block": {},
9 #   "golden bowl": {},
10 #   "red bowl": {},
11 # }
12 [User Query 1] put the white block in the golden bowl
13 [Gold Code]
14   put_first_on_second("white block", "golden bowl")
15 [User Query 2] put the cyan block on the green block
16 [Gold Code]
17   put_first_on_second("cyan block", "green block")
18 [User Query 3] put the orange block in the empty bowl
19 [Gold Code]
20   put_first_on_second("orange block", "red bowl")
21 [User Query 4] put the block in the golden bowl on the block in the red bowl
22 [Gold Code]
23   put_first_on_second("white block", "orange block")
24 [User Query 5] is the green block in a higher position than the white block
25 [Gold Code]
26   say("no")
27 [User Query 6] is the green block right above the orange block
28 [Gold Code]
29   say("no")

```

---

Listing B.2: Example evaluation episode (Disinfection domain). The query in [blue](#) requires temporal reasoning.

---

```

1 [Initial State]
2 # state = {
3 #   "objects": ["teal block", "black block", "cyan block", "blue block", "tan bowl", "
  disinfectant"],
4 #   "relations": [],
5 #   "disinfectant": {"contains": []},
6 #   "teal block": {},
7 #   "black block": {},
8 #   "cyan block": {},
9 #   "blue block": {},
10 #   "tan bowl": {}
11 # }
12 [User Query 1] the teal block and the black block are dirty.
13 [User Query 2] The other blocks are clean
14 [User Query 3] Woops, somebody accidentally polluted the cyan block and the blue block
15 [User Query 4] Put the cyan block on the teal block
16 [Gold Code]
17     put_first_on_second("cyan block", "teal block")
18 [User Query 5] Put the blue block and the black block in the disinfectant
19 [Gold Code]
20     put_first_on_second("blue block", "disinfectant")
21     put_first_on_second("black block", "disinfectant")
22 [User Query 6] Put the blue block in the tan bowl
23 [Gold Code]
24     put_first_on_second("blue block", "tan bowl")
25 [User Query 7] Put the blue block on the table.
26 [Gold Code]
27     put_first_on_second("blue block", "table")
28 [User Query 8] Put the black block on the blue block
29 [Gold Code]
30     put_first_on_second("black block", "blue block")
31 [User Query 9] Put all the dirty blocks on the table.
32 [Gold Code]
33     put_first_on_second("teal block", "table")
34     put_first_on_second("cyan block", "table")

```

---

Listing B.3: Example evaluation episode (Relative Weight domain). The query in [blue](#) requires temporal reasoning.

---

```

1 [Initial State]
2 # state = {
3 #   "objects": ["black block", "orange block", "green block", "red block", "gray bowl",
4 #   "blue bowl"],
5 #   "relations": [],
6 #   "black block": {},
7 #   "orange block": {},
8 #   "green block": {},
9 #   "red block": {},
10 #   "gray bowl": {},
11 #   "blue bowl": {},
12 # }
13 [User Query 1] The black block is twice the weight of the green block
14 [User Query 2] Put the orange block in the gray bowl
15 [Gold Code]
16   put_first_on_second("orange block", "gray bowl")
17 [User Query 3] The red block is twice the weight of the orange block
18 [User Query 4] The red block has the same weight of the black block
19 [User Query 5] Put the red block in the gray bowl
20 [Gold Code]
21   put_first_on_second("red block", "gray bowl")
22 [User Query 6] Put blocks in the blue bowl so that their total weight becomes identical
23   to what is in the gray bowl
24 [Gold Code]
25   put_first_on_second("black block", "blue bowl")
26   put_first_on_second("green block", "blue bowl")

```

---

Listing B.4: Prompt for Code-as-Policies (Disinfection domain).

---

```

1 In the following, the robot deals with dirty and clean blocks. A clean block becomes
  dirty when it touches another dirty block. This includes when a dirty block is stacked on
  top of a clean block, and also when a dirty block is right under a clean block. The
  table, bowls and the robot gripper are protected from any dirtiness, so they stay clean
  forever. When a dirty block is put into the disinfecter, it becomes clean immediately.
2
3 Instruction:
4 Aside from the built-in python functions and statements, the robot can only run the
  following functions:
5 `put_first_on_second`, `say` and `noop`.
6
7 Each code is carefully designed by professionals to meet all of these requirements.
8 ===
9 # objects = ["cyan block", "yellow block", "brown block", "purple block", "blue block", "
  green bowl", "red bowl", "disinfecter"]
10 # query: The cyan block and purple block are dirty
11 noop()
12 # query: The other blocks are clean
13 noop()
14 # query: Put the cyan block on the yellow block
15 put_first_on_second("cyan block", "yellow block")
16 # query: Put the brown block in the green bowl
17 put_first_on_second("brown block", "green bowl")
18 # query: Woops, somebody took out the brown block and dropped it on a dirty area
19 noop()
20 # query: Pick the cyan block and put it on the table
21 put_first_on_second("cyan block", "table")
22 # query: Move the yellow block into the disinfecter
23 put_first_on_second("yellow block", "disinfecter")
24 # query: Place all the clean blocks in the green bowl
25 # THINK: The clean blocks are yellow block and purple block
26 put_first_on_second("blue block", "green bowl")
27 put_first_on_second("yellow block", "green bowl")
28 # query: Put the cyan and purple block in the disinfecter
29 put_first_on_second("cyan block", "disinfecter")
30 put_first_on_second("purple block", "disinfecter")
31 # query: Put the dirty blocks in the red bowl
32 # THINK: The only dirty block is the brown block
33 put_first_on_second("brown block", "red bowl")
34 # query: Pick the blue block and put it on the table
35 put_first_on_second("blue block", "table")
36 # query: Put the yellow block on the table
37 put_first_on_second("yellow block", "table")
38 # query: Put the brown block in the green bowl
39 put_first_on_second("cyan block", "green bowl")
40 # query: How many blocks are in the red bowl
41 # THINK: the red bowl is empty
42 say("There is no block in the red bowl")
43 ===

```

---

Listing B.5: Prompt for Statler (Disinfection domain). To avoid getting cluttered, many lines are omitted (denoted with “omitted”). Please check our repository for full version.

---

```

1 In the following, the robot deals with dirty and clean blocks. A clean block becomes
  dirty when it touches another dirty block. This includes when a dirty block is stacked on
  top of a clean block, and also when a dirty block is right under a clean block. The
  table, bowls and the robot gripper are protected from any dirtiness, so they stay clean
  forever. When a dirty block is put into the disinfecter, it becomes clean immediately.
2
3 Instruction:
4 Aside from the built-in python functions and statements, the robot can only run the
  following functions:
5 `put_first_on_second`, `say` and `update_wm`.
6
7 Each code is carefully designed by professionals to meet all of these requirements.
8 ===
9 # state = {
10 #   "objects": ["cyan block", "yellow block", "brown block", "purple block", "blue
    block", "green bowl", "red bowl", "disinfecter"],
11 #   "relations": [],
12 #   "disinfecter": {"contains": []},
13 #   "cyan block": {},
14 #   "yellow block": {},
15 #   "brown block": {},
16 #   "purple block": {},
17 #   "blue block": {},
18 #   "green bowl": {},
19 #   "red bowl": {}
20 # }
21 # query: The cyan block and purple block are dirty
22 update_wm("The cyan and purple blocks are dirty")
23 # state = { ... omitted ...
24 # query: The other blocks are clean
25 update_wm("The brown block, yellow block, and blue block are clean")
26 # state = { ... omitted ...
27 # query: Put the cyan block on the yellow block
28 put_first_on_second("cyan block", "yellow block")
29 update_wm("Put the cyan block on the yellow block.")
30 # state = {
31 #   "objects": ["cyan block", "yellow block", "brown block", "purple block", "blue
    block", "green bowl", "red bowl", "disinfecter"],
32 #   "relations": [["cyan block is on yellow block"]],
33 #   "disinfecter": {"contains": []},
34 #   "cyan block": {"is": ["dirty"]},
35 #   "yellow block": {"is": ["dirty"]},
36 #   "brown block": {"is": ["clean"]},
37 #   "purple block": {"is": ["dirty"]},
38 #   "blue block": {"is": ["clean"]},
39 #   "green bowl": {},
40 #   "red bowl": {}
41 # }
42 # query: Put the brown block in the green bowl
43 put_first_on_second("brown block", "green bowl")
44 update_wm("Put the brown block in the green bowl")
45 # state = { ... omitted ...

```

```

46 # query: Woops, somebody took out the brown block and dropped it on a dirty area
47 update_wm("Put the brown block on the table, and make it dirty")
48 # state = { ... omitted ...
49 # query: Pick the cyan block and put it on the table
50 put_first_on_second("cyan block", "table")
51 update_wm("Put the cyan block on the table")
52 # state = { ... omitted ...
53 # query: Move the yellow block into the disinfecter
54 put_first_on_second("yellow block", "disinfecter")
55 update_wm("Put the yellow block in the disinfecter")
56 # state = { ... omitted ...
57 ..... omitted .....
58 # query: Put the brown block in the green bowl
59 put_first_on_second("cyan block", "green bowl")
60 update_wm("Put the cyan block in the green bowl")
61 # state = { ... omitted ...
62 # query: How many blocks are in the red bowl
63 # THINK: the red bowl is empty
64 say("There is no block in the red bowl")
65 # state = { ... omitted ...
66 ===

```

---

## B.3 Example Failure Cases

We manually looked into examples where Statler fails. Although providing insights on when and why the model fails would be helpful, we find that in many cases it is not trivial to interpret the model’s mistakes. In this section, we show three episodes (one per each domain) that the model failed, just to give some ideas on its failure modes.

### B.3.1 Pick-and-Place

The full episode, including the generated state and code, is shown in Listing B.6. Given the fifth user query, "Is the green block in a higher position than the white block?" The current state correctly identifies the positions of both the white and green blocks (i.e., the white block is on the orange block, which is inside the red bowl, and the green block is on the table). However, with this information correctly absorbed as shown in its thinking process, the state reader still fails to recognize that the green block is in a lower position.

### B.3.2 Disinfection

The full episode, including the generated state and code, is shown in Listing B.7. Given the eighth user query "Put the black block on the blue block",

Listing B.6: Generated states and code on an evaluation episode (Pick-and-Place domain). Incorrect portions are highlighted in red.

---

```

1 [Initial State]
2 # state = {
3 #   "objects": ["green block", "orange block", "white block", "cyan block", "golden
4 #     bowl", "red bowl"],
5 #   "relations": [],
6 #   "green block": {},
7 #   "orange block": {},
8 #   "white block": {},
9 #   "cyan block": {},
10 #   "golden bowl": {},
11 #   "red bowl": {},
12 # }
13 ..... omitted .....
14 # state = {
15 #   "objects": ["green block", "orange block", "white block", "cyan block", "golden
16 #     bowl", "red bowl"],
17 #   "relations": ["cyan block is on green block", "white block is on orange block"],
18 #   "green block": {},
19 #   "orange block": {},
20 #   "white block": {},
21 #   "cyan block": {},
22 #   "golden bowl": {},
23 #   "red bowl": {"contains": ["orange block"]},
24 # }
25 [User Query 5] is the green block in a higher position than the white block
26 [Generated Code]
27 # THINK: The cyan block is on the green block which is on the table
28 # THINK: The white block is on the orange block which is in the red bowl
29 # THINK: Thus the green block is in a higher position
30 say("yes")
31 [Gold Code]
32 say("no (white block is on the orange block is in the red bowl, green block is on the
33   table)")
34 ..... omitted .....

```

---



even though the current state correctly recognizes both the black and blue blocks to be *clean*, the generated next state marks them “dirty” by mistake (highlighted in red). This error by the state writer causes the failure of the code generation at the next step.

### B.3.3 Relative Weight Reasoning

The full episode, including the generated state and code, is in Listing B.8. The model succeeds up to user query 5. Given the next user query “Put blocks in the blue bowl so that their total weight becomes identical to what is in the gray bowl”, the reasoning process that we can observe from # THINK: comments on the generated code contains a hallucination (highlighted in red). While the current state correctly identifies that the blue bowl is empty, the model *thinks* the green block is in it, and as a result fails to move it over to the blue bowl.

Listing B.7: Generated states and code on an evaluation episode (Disinfection domain). Incorrect portions are highlighted in red.

---

```

1 [Initial State]
2 # state = {
3 #   "objects": ["teal block", "black block", "cyan block", "blue block", "tan bowl", "
  disinfectant"],
4 #   "relations": [],
5 #   "disinfectant": {"contains": []},
6 #   "teal block": {},
7 #   "black block": {},
8 #   "cyan block": {},
9 #   "blue block": {},
10 #   "tan bowl": {}
11 # }
12 ..... omitted .....
13 # state = {
14 #   "objects": ["teal block", "black block", "cyan block", "blue block", "tan bowl", "
  disinfectant"],
15 #   "relations": [{"cyan block is on teal block"}],
16 #   "disinfectant": {"contains": ["black block"]},
17 #   "teal block": {"is": ["dirty"]},
18 #   "black block": {"is": ["clean"]},
19 #   "cyan block": {"is": ["dirty"]},
20 #   "blue block": {"is": ["clean"]},
21 #   "tan bowl": {}
22 # }
23 [User Query 8] Put the black block on the blue block
24 [Generated Code]
25   put_first_on_second("black block", "blue block")
26   update_wm("Put the black block on the blue block")
27 [Gold Code]
28   put_first_on_second("black block", "blue block")
29 [Generated State]
30 # state = {
31 #   "objects": ["teal block", "black block", "cyan block", "blue block", "tan bowl", "
  disinfectant"],
32 #   "relations": [{"cyan block is on teal block"}, {"black block is on blue block"}],
33 #   "disinfectant": {"contains": []},
34 #   "teal block": {"is": ["dirty"]},
35 #   "black block": {"is": ["dirty"]},
36 #   "cyan block": {"is": ["dirty"]},
37 #   "blue block": {"is": ["dirty"]},
38 #   "tan bowl": {}
39 # }
40 [User Query 9] Put all the dirty blocks on the table.
41 [Generated Code]
42   put_first_on_second("teal block", "table")
43   put_first_on_second("black block", "table")
44   put_first_on_second("cyan block", "table")
45   put_first_on_second("blue block", "table")
46   update_wm("Put all the dirty blocks on the table")
47 [Gold Code]
48   put_first_on_second("teal block", "table")
49   put_first_on_second("cyan block", "table")

```

---

Listing B.8: Generated states and code on an evaluation episode (Relative Weight domain). Incorrect portions are highlighted in red.

---

```

1 [Initial State]
2 # world_state = {
3 #     "objects": ["black block", "orange block", green block", "red block", "gray bowl",
4 #     "blue bowl"],
5 #     "relations": [],
6 #     "black block": {},
7 #     "orange block": {},
8 #     "green block": {},
9 #     "red block": {},
10 #     "gray bowl": {},
11 #     "blue bowl": {},
12 # }
13 ..... omitted .....
14 [Generated State]
15 # state = {
16 #     "objects": ["black block", "orange block", green block", "red block", "gray bowl",
17 #     "blue bowl"],
18 #     "relations": [],
19 #     "black block": {"weight": red_block.weight},
20 #     "orange block": {"weight": red_block.weight / 2},
21 #     "green block": {"weight": black_block.weight / 2},
22 #     "red block": {"weight": black_block.weight},
23 #     "gray bowl": {"contains": ["orange block", "red block"]},
24 #     "blue bowl": {},
25 # }
26 [User Query 6] Put blocks in the blue bowl so that their total weight becomes identical
27 to what is in the gray bowl
28 [Generated Code]
29 # THINK: In the gray bowl, we have orange block and red block
30 # THINK: In the blue bowl, we have green block
31 # THINK: The orange block has half the weight of the red block
32 # THINK: The green block has half the weight of the black block
33 # THINK: We need to place a block that has `red_block.weight / 2`
34 # THINK: The block that has the weight `red_block.weight / 2` is black block
35 put_first_on_second("black block", "blue bowl")
36
37 update_wm("Put the black block in the blue bowl")
38 [Gold Code]
39 put_first_on_second("black block", "blue bowl")
40 put_first_on_second("green block", "blue bowl")

```

---