GarmentLab: A Unified Simulation and Benchmark for Garment Manipulation

Haoran Lu[1,2*] **Ruihai Wu**[1*] **Yitong Li**[1,3*]
Sijie Li[1,2] Ziyu Zhu[1,2] Chuanruo Ning[1] Yan Shen[1]
Longzan Luo[1,2] Yuanpei Chen[1] Hao Dong [1]
[1]CFCS, School of CS, PKU [2]School of EECS, PKU [3]Weiyang College, THU

Abstract

Manipulating garments and fabrics has long been a critical endeavor in the development of home-assistant robots. However, due to complex dynamics and topological structures, garment manipulations pose significant challenges. Recent successes in reinforcement learning and vision-based methods offer promising avenues for learning garment manipulation. Nevertheless, these approaches are severely constrained by current benchmarks, which offer limited diversity of tasks and unrealistic simulation behavior. Therefore, we present **GarmentLab**, a content-rich benchmark and realistic simulation designed for deformable object and garment manipulation. Our benchmark encompasses a diverse range of garment types, robotic systems and manipulators. The abundant tasks in the benchmark further explores of the interactions between garments, deformable objects, rigid bodies, fluids, and human body. Moreover, by incorporating multiple simulation methods such as FEM and PBD, along with our proposed sim-to-real algorithms and real-world benchmark, we aim to significantly narrow the sim-to-real gap. We evaluate state-of-the-art vision methods, reinforcement learning, and imitation learning approaches on these tasks, highlighting the challenges faced by current algorithms, notably their limited generalization capabilities. Our proposed open-source environments and comprehensive analysis show promising boost to future research in garment manipulation by unlocking the full potential of these methods. We guarantee that we will open-source our code as soon as possible. You can watch the videos in supplementary files to learn more about the details of our work. Our project page is available at: `https://garmentlab.github.io/`

1 Introduction

The next-generation assistant robots should possess not only the abilities to separately manipulate a wide variety of objects, including rigid, articulated[59], and deformable objects[58], but also the capability to leverage interactions between those physical media, including flow and fluids, in order to assist humans[39]. Among various daily tasks [69, 59, 56], garment manipulation stands out as one of the most challenging, crucial, and extensively discussed tasks in the robotics and computer vision, due to its demanding requirements for understanding dynamic properties of physical instances and interactions between them. For instance, washing clothes entails the interaction between garments and fluids, while dressing up requires collaboration between robots and humans.

Garment manipulation tasks mainly presents three challenges. Firstly, each individual garment possesses nearly infinite states and exhibits complex kinematic and dynamic properties. *Therefore, it is crucial for models to comprehend the various self-deform states of garments, which usually requires*

*Equal contribution.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-0379

Figure 1: **GarmentLab** provides realistic simulation for diverse garments with different physical propoerties, benchmarking various novel garment manipulation tasks in both simulation and the real world.

large amount of training data [17, 6] **(C1)**. Secondly, garment manipulation involves interactions with various types of objects, including rigid (*e.g.*, clothes hanger) and articulated (*e.g.*, wardrobe) objects, as well as fluids and human body. *Consequently, enabling models to understand these interactions across diverse physical media presents great significance* **(C2)**. Finally, considering that strategies for manipulating garments are often highly complex, and visual perception of garments is more challenging due to their diverse states and patterns, *manipulating garments faces a greater sim2real gap [63, 29]* **(C3)**.

Training a powerful agent capable of overcoming these challenges necessitates a vast amount of data encompassing robot-object interactions. However, directly collecting data from the real world is impractical. Thus, researchers have long pursued benchmarks for garment manipulation [30, 4, 6, 64]. Current deformable simulations suffer from drawbacks such as missing garment meshes [30]. Additionally, they offer a limited range of tasks, hindering further research endeavors.

Therefore, we present **GarmentLab** (Figure 1), a unified environment and benchmark for garment manipulation. **GarmentLab** has three novel components to satisfy the demands for diversity and realism: The powerful **GarmentLab Engine**, which is built upon Omniverse Isaac Sim [71] and supports a variety of physical simulation methods. The simulator not only supports Position-Based-Dynamic (PBD) [3], Finite-Element-Method (FEM) [11], to simulate garments, fluid and deformable objects but also makes integration with ROS [42] to provide an efficient teleoperation pipeline for data collection. **GarmentLab Assets** is a large-scale indoor dataset comprising 1) garments models covering 11 categories of daily garments from ClothesNet [70] 2) various kinds of robot end-effector including gripper, suction and dexterous hands. 3) high-quality 3D assets including 20 scenes and 9000+ object models from ShapeNet [7]. Based on realistic simulation and rich assets, we propose **GarmentLab Benchmark** containing 20 tasks divided into 5 groups to evaluate state-of-the-art vision-based and reinforcement learning based algorithm.

To tackle above challenges, our environment has three characteristics:1) **Efficient**. Garment manipulation involves nearly infinite object state and action spaces, requiring substantial data for models to understand garment structure and deformation. To meet this demand, our highly parallelized GPU-based simulator provides a significant training advantage. Larger batch sizes enhance RL-based algorithms [33], while faster data collection speeds reduce training time for perception-based algorithms (tackling **C1**). 2) **Rich**. The richness of our simulator can be categorized into two aspects: the diversity of **simulation content** offered by GarmentLab Assets and the depth of **physical interaction** facilitated by GarmentLab Engine. We emphasize multi-physics simulation, encompassing rigid-articulated, deformable-garment, fluid dynamics, and flow, along with their interactions. This focus is vital for training agents capable of comprehending real-world physical properties [48] (tackling **C2**). You can refer to videos in supplementary material for our simulation effects. 3) **Real**.

https://doi.org/10.52202/079017-0379

Table 1: **Comparisons with Other Deformable Object Environments.**

	Multi-Camera	Scenes	Robot	Sim2real	Garment	Fluid	Rigid Body	Articulated	Human	FEM Objects	Flow	RealBenchmark	Physics	Thermal(stream/fire)	Mobile Task	Dexterous Task	Rendering	Pipeline	Area
Softgym[30]	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	R	GPU	Manipulation
Orbit[34]	✓	✓	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	RT	GPU	Industrial
Sapien[62]	✓	✗	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	R	CPU	Manipulation
Habitat 3.0[39]	✗	✓	✓	✗	✗	✗	✓	✗	✓	✗	✗	✗	✗	✗	✗	✗	R	CPU	Navigation
Behavior[28]	✗	✓	✓	✗	✓	✗	✓	✗	✗	✓	✗	✗	✗	✓	✗	✗	RT	GPU	Manipulation
Mujoco[53]	✓	✗	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	R	CPU	Manipulation
Pybullet[12]	✓	✗	✓	✗	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	R	CPU	Manipulation
RLBench[23]	✓	✗	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	R	CPU	Manipulation
Gibson [60]	✓	✓	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✓	✗	R	CPU	Navigation
DeformableRavens[47]	✓	✗	✓	✗	✗	✗	✓	✗	✗	✓	✗	✗	✓	✗	✗	✗	R	CPU	Manipulation
GarmentLab	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	**RT**	**GPU**	Manipulation

As the sim-to-real gap emerges as the main obstacle in developing embodied agents, GarmentLab Engine surpasses Omniverse capabilities by providing mature sim-to-real algorithms, such as Teleoperation [41] utilized in the RL field, and the Visual Sim-Real Alignment Algorithm employed in perception algorithms. We also make integration with ROS [42] and MoveIt [10], which is beneficial for narrowing sim2real gap by introducing real-world robot motions into simulation (tackling **C3**).

Our benchmark experiments highlight the significant challenges current algorithms face, even with seemingly simple tasks like unfolding. These difficulties arise from a lack of understanding of physical interactions and the complexities of high-dimensional states. Vision-based algorithms demonstrate limited generalization, with performance strongly affected by the initial state of objects. RL-based algorithms also encounter difficulties with tasks requiring long-horizon planning. These insights offer valuable guidance for improving methods for garment and deformable object manipulation.

In summary, we have made the following contributions in **GarmentLab**:

- We propose **GarmentLab Environment**, a **realistic and rich environment** for garment manipulation, featuring diverse simulation methods, assets, object physics and multi-material interactions.
- Based on GarmentLab Environment, we propose **GarmentLab Benchmark**, benchmarking a large variety of garment manipulation tasks, and providing the first real-world garment manipulation benchmark that can be reproduced internationally.
- We **integrate different sim2real methods** into GarmentLab, providing solutions to narrowing the sim2real and further facilitating the real-world applications.
- **Extensive experiments and detailed analyses** of different types of garment manipulation algorithms facilitate and enlight future research on garment manipulation.

2 Related Work

Garment and Deformable Object Benchmarks. Current deformable object environments [30, 61, 63] usually support only one simulation method (*e.g.*, PBD or FEM), limiting the types of simulated objects and interactions. Besides, most environments are CPU-based [62, 63, 12], severely limiting parallel capabilities and often exhibiting a huge sim-to-real gap due to the absence of comprehensive sim-to-real algorithm designs. In contrast, as a GPU-based simulator, GarmentLab provides diverse 3D meshes and supports various simulation techniques. We further integrate ROS and combine it with our carefully designed sim-to-real pipeline, offering a more comprehensive solution for researchers. Detailed comparisons between our environment and others can be found in Table 1 and Appendix C

Garment and Deformable Object Manipulation. Although current efforts excel at specific tasks such as folding[63, 1, 57] and unfolding[17, 6], many real-world tasks are long-horizon and involve interactions between various physical media. While many studies have potential to tackle these problems[48, 29], they are hindered by the lack of a mature simulation platform capable of supporting such diverse and complicated extensions. Furthermore, while current research predominantly emphasizes gripper manipulation tasks[2, 65], we introduce tasks involving suction, dexterous hands, and mobile robots. We believe GarmentLab will make a unique and valuable contribution to the robotics community by providing a new platform for developing garment manipulation algorithms and significantly expanding the scope of existing methods.

https://doi.org/10.52202/079017-0379

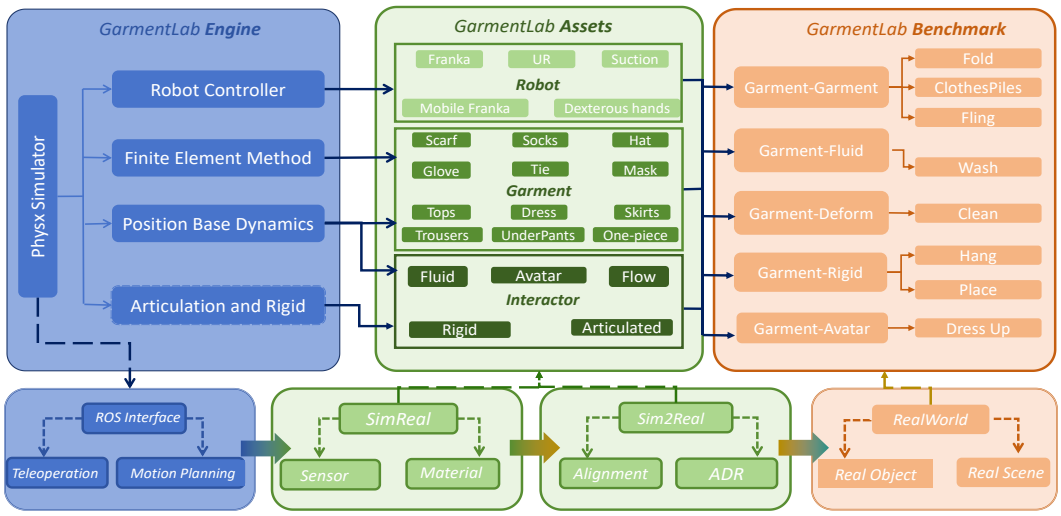


Figure 2: **The Architecture of GarmentLab. (Left)** Built on PhysX5, our environment supports various simulation methods. **(Middle)** Our environment can deliver realistic simulations of diverse robots, garments, and interactions between multiple physics media. **(Right)** Subsequently, we can utilize these assets to construct tasks across various categories. **(Bottom)** The framework supports real-world deployment.

3 GarmentLab Environment

GarmentLab aims to integrate state-of-the-art physical simulation methods, modern graphics rendering engines, and user-friendly robotic interfaces into a unified framework (Figure 2). Below we will first introduce *GarmentLab Engine* (Section 3.1) and *GarmentLab Asset* (Section 3.2) to show our diversity in function and objects. As we especially focus on the exploration of multiple physical simulation methods and interaction between them, we will introduce *GarmentLab Physics* in Section 3.3. In section 4 We will talk about our novel-proposed tasks.

3.1 GarmentLab Engine

Built on NVIDIA's IsaacSim[71], GarmentLab offers a highly-paralleled data collection pipeline, realistic rendering, support for various sensors, and integration with Robot Operating System (ROS) [42].

Data Pipeline. Data pipeline mainly consists of two components: Visual Data System and RL-Training System. Visual System provides both RGB-D observations and ground-truth semantic label including 2D and 3D bounding box, normals and instance segmentation. Based on IsaacGym, RL System can establish multiple agents on GPU at the same time for efficient training.

Rendering. GarmentLab supports multiple camera angles, such as eye-on-hand and eye-on-base perspectives, unlike the single-camera setups of past works [30, 61],which employing naive OpenGL framework[49]. Additionally, it utilizes GPU-enabled ray tracing for rendering, which enhances realism and challenge by creating more realistic shadows and lighting [51], thus reducing the sim2real gap and improving the performance of visual algorithms and mobile navigation tasks.

ROS. ROS[42] is a generic and widely-used framework for building robot applications. We use ROS to align robot in realworld and the simulation, please refer to Section 6.2 for detailed. Also, although IsaacSim provides traditional Inverse-Kinematic[36] and RMPFlow control[27], we also provide MoveIt framework[10] for motion planning, which is more widely used in the real world.

Sensor. In addition to RGB-D observations, auxiliary observations can be accessed, such as robot joints, cloth particles and object poses. They are required in common RL framework and teacher-student network[15]. Other Omniverse sensors (e.g., tactile, contact-report) could also be available.

https://doi.org/10.52202/079017-0379

3.2 GarmentLab Assets

GarmentLab Asset compiles simulation content from a variety of state-of-the-art datasets, integrating individual meshes or URDF files into complete, simulation-ready scenes with robots and sensors. We employ Universal Scene Description files to store all assets with attributes, including physics, semantics, and rendering properties. Key components along with their sources and categories are shown in Table 2. More details about each asset type are provided in Appendix A.

Table 2: **Key Components of GarmentLab Assets.**

Asset Type	Sources	Categories
Garment and Cloth	ClothesNet	Hats, Ties, Masks, Gloves, Socks, Dishcloths, ...
Rigid and Articulated	ShapeNet, PartNet, YCB, PartNet-Mobility	Chairs, Boxes, Washing Machines, Storage Furniture, ...
Robot	-	Franka, UR5, RidgebackFranka, ShadowHands
Human Model	Omniverse HumanModel	-
Materials	Omniverse Base Material	Fabric, Carpet, Leather, Silk, Cotton, ...

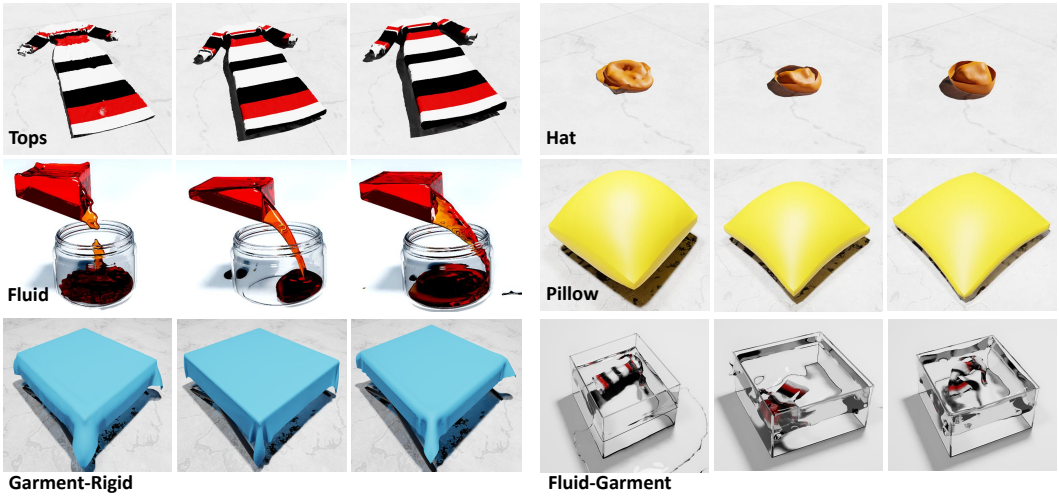


Figure 3: **GarmentLab Physics.** GarmentLab explores the potential of different simulation methods, and provides different physical parameters, modeling the distinct properties of different materials in the real world.

3.3 GarmentLab Physics

Simulation Method. To ensure physically realistic simulation, we use tailored methods for different objects based on their physical characteristics. For **large garments and fluid**, we use Postion-Based Dynamics (PBD)[3]. For **small elastic garments** like gloves and socks, and everyday objects like toys and sponges, we apply Finite Element Method (FEM)[11]. **Human simulation** involves articulated skeletons with rotational joints and a surface skin mesh for high-fidelity rendering. **Robot simulation** utilizes PhysX articulation system for precise force control, P-D control, and inverse dynamics. Unlike previous works that rely on a single simulation method, GarmentLab provides platforms for exploring dynamics and kinematics of various objects and the coupling and interactions among them.
Diverse Physics Parameters. To fully exploit the potential of various simulation methods and make garment simulation more diverse and realistic, GarmentLab provides various physics parameter configurations. For example, as cloth is modeled as a grid of particles, altering parameters such as particle size and stiffness will change garment physical behaviors. Likewise, as depicted in Figure 3, diverse physical material parameters are assigned to diverse objects. These parameters encompass, but are not limited to, surface tension and cohesion for fluids, friction for rigid objects, and modulus. It is worth noting that parameters influencing the interaction between different objects, including contact offset and reset offset, are also adjusted. For further details, please consult Appendix D.

https://doi.org/10.52202/079017-0379

4 GarmentLab Benchmark

GarmentLab Benchmark is motivated by the abilities that an intelligent manipulator agent should possess, including (1) understanding the physics of object interactions, (2) generating accurate action sequences for long-horizon, complex tasks, and (3) transferring this knowledge to the real world. To test these abilities, we classified tasks into five categories based on physical interactions. We also proposed several complex, long-horizon tasks to advance future research. The demos of tasks and real-world experiment are shown in figure 4.

Task Categories. To fully exploit the model's capability in understanding physical interactions and conduct comprehensive evaluations of current algorithms, we categorize 20 tasks into 5 groups. The example tasks and corresponding categories are shown in Table 3. Examples of task sequences are provided in Figure 4. You can refer to Appendix G to get more details.

Long-Horizon Tasks. With the advancement of robotics, there is a growing focus on completing long-horizon tasks, which integrate skills tasks such as 3D perception, manipulation and navigation. Thus we propose several long-horizon garment manipulation tasks, including *organizing clothes*, *wash clothes*, *make up tables* and *dress up*. These tasks go beyond simply executing subtasks in sequence, as they require holistically planning how to accomplish the task based on the environment. During the execution, the algorithm needs to consider the positioning of the operation, the placement location, and carefully plan the path to avoid collisions. More analysis are shown in Section 7.

Table 3: **Task Categories of GarmentLab Benchmark.**

Task Type	Focus	Example
Garment-Garment	fundamental garment manipulation	fold, unfold, clothes piles retrieval...
Garment-Fluid	interaction between garments and fluids as well as flow,	Washing clothes, Drying clothes with a hairdryer...
Garment-FEMObjects	manipulating FEM Objects	Packing clothes, Dexterous grasp plush toy
Garment-Rigid	common interactions between clothing and rigid bodies	Hanging, Putting clothes into washing machine
Garment-Avatar	collaboration with human	Putting a scarf, Dress a person in T-shirt

5 Real-World Benchmark

Real-world benchmark is crucial for not only evaluating the real-world performance of different methods, but also providing a standardized platform for researchers to reproduce and exchange methods. With the existence of real-world dataset or benchmarks for rigid [5], articulated [31] objects and furnitures [18], we introduce the first real-world benchmark for deformable objects and garments.

Unlike rigid or articulated objects that can be 3D-printed from CAD files, deformable objects are usually purchased without CAD files. Easily influenced by external forces, it is difficult to accurately model garments directly using traditional multi-camera calibration and surface reconstruction methods. Therefore, we use commercial scanning devices with lasers and light for mesh scanning.

Selected objects cover diverse garments (tops, trousers, socks, hats), plush toys, household items (bags, clutches), and cleaning supplies. They are primarily selected from well-known international brands for durability and accessibility. To ensure variety, objects have different shapes, sizes, transparencies, deformabilities, and textures. For instance, our dataset features various tops made from materials like assault jackets, down jackets, shirts, and vests, with a wide range of physical attributes.

Additionally, we provide semantic human annotations for object part masks and key points, supporting dexterous manipulation such as grasping specific parts and object tracking using key points. Following YCB[5], we present a systematic approach for defining manipulation protocols and benchmarks. These protocols specify the experimental setup for each task and provide procedural guidelines. A comprehensive description of the real-world benchmark is provided in Appendix F.

6 Sim2Real Framework

Transferring models from simulation to reality is crucial yet challenging. GarmentLab paves way to realistic application by integrating methods for mitigating vision (Sec. 6.1) and action (Sec. 6.2) gap.

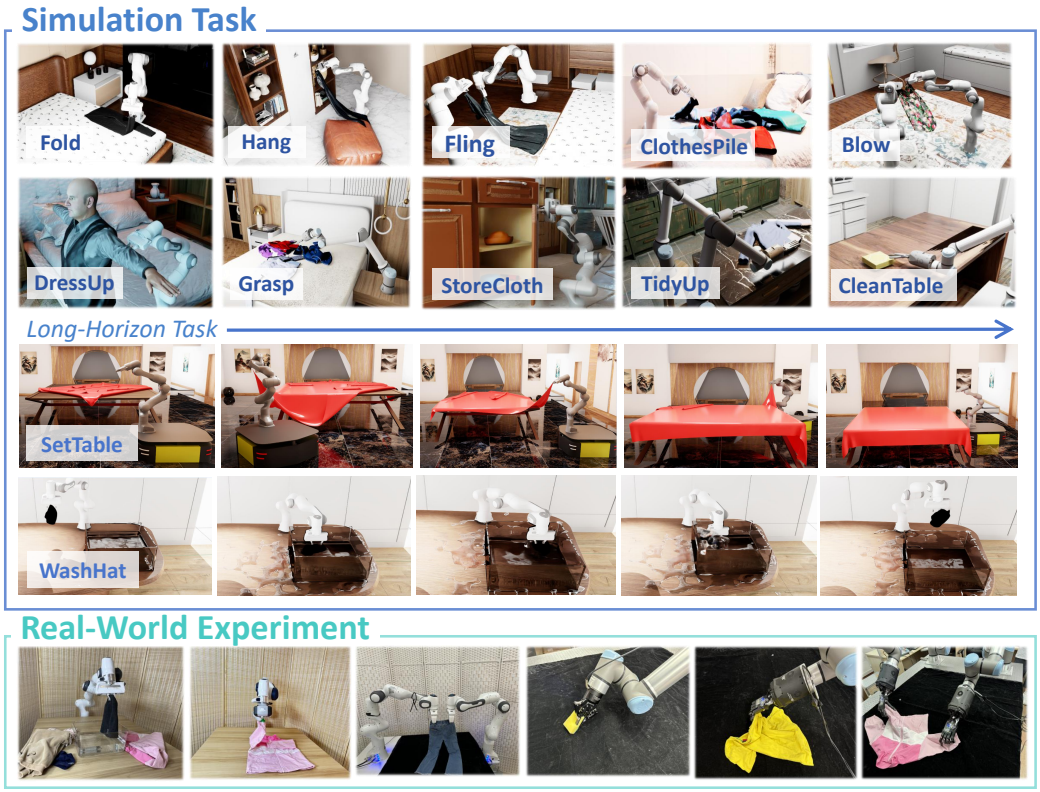


Figure 4: **Diverse Tasks of GarmentLab Benchmark.** We introduced 20 garment and deformable manipulation tasks including complicated long-horizon tasks. The last row shows the execution of these tasks in the real world.

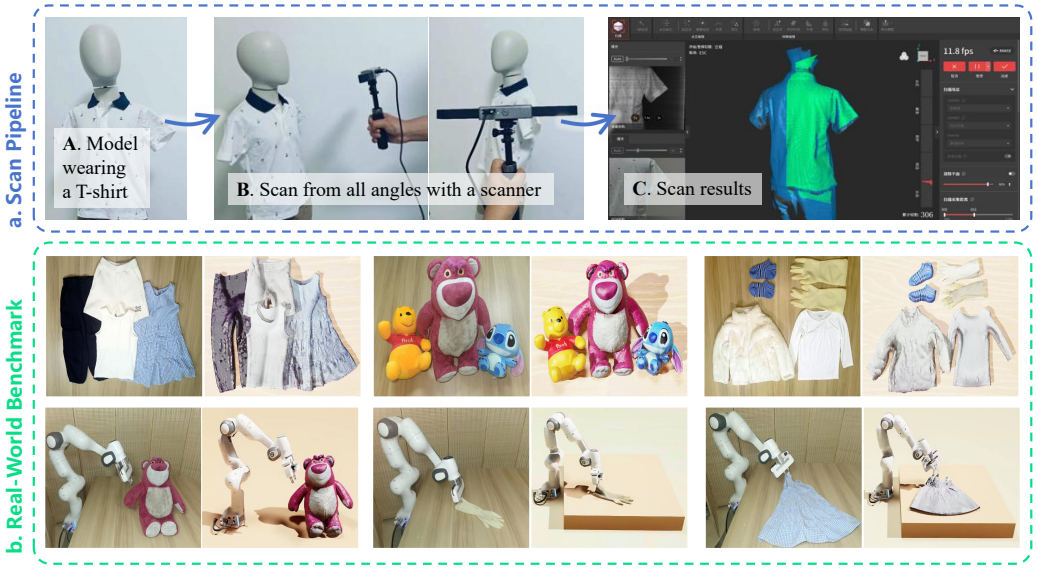


Figure 5: **Real-World Benchmark.** Part a demonstrates the whole pipeline of converting real-world objects into simulation assets. Part b demonstrates the performance of different categories of objects in both simulation and the real world (the first row), and the results of these objects being manipulated by the robot (the second row).

https://doi.org/10.52202/079017-0379

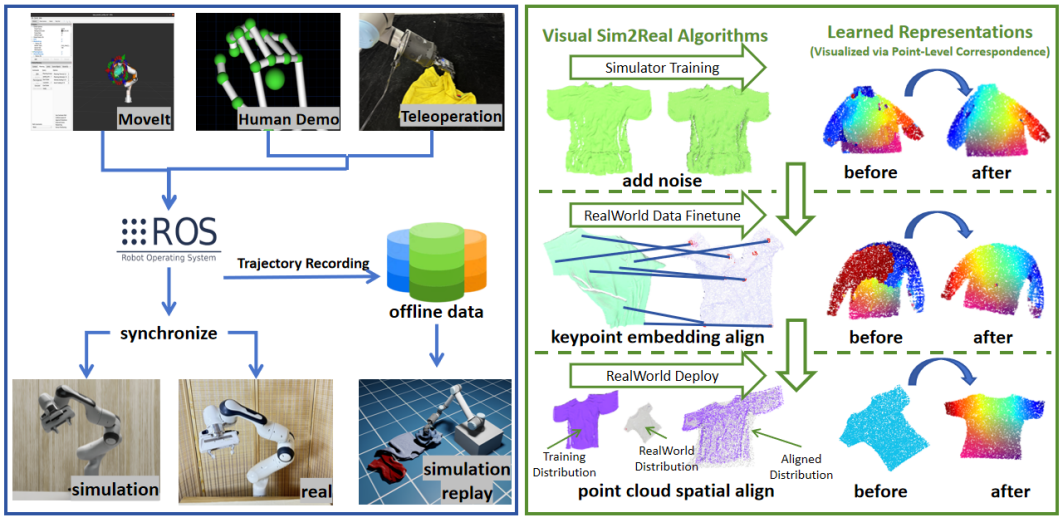


Figure 6: **Sim2Real Framework.** On the left, we highlight our MoveIt and teleoperation pipeline, a lightweight and easy-to-deploy system built using ROS. On the right, we present our three proposed visual sim-to-real algorithms, demonstrating a significant improvement in model performance after deploying these algorithms.

6.1 Sim-Real Vision Alignment

GarmentLab intergrates several automated and self-supervised sim2real methods, and have verified their effectiveness by predicting dense visual correspondence for manipulation [57] (Figure 6, Right), with quantitative manipulation success rate in Table 6.

Keypoint Embedding Alignment. Aligning corresponding skeleton point representations can mitigate representation gap between point cloud in simulation and the real world [57]. By attaching markers to skeleton points and enabling robot to perform self-play, we obtain ground-truth keypoint pairs and employ InfoNCE [25]to align corresponding point representations. Shown in Figure 6, the alignment adapts representations to the real-world distribution. Appendix E shows more details.
Noisy Observation. Adding noise to point cloud for training can be very effective for sim2real transfer [16]. As shown in Figure 6, initial query results had many errors. By adding noise during training, our model became more robust, leading to smoother and more accurate representations.
Point Cloud Alignment. We propose aligning point clouds by optimizing an affine matrix, using chamfer distance as the loss function. As shown in Figure 6, the model initially predicts incorrect results, even for flat surfaces. However, after alignment, it successfully predicts accurate results.

6.2 Real-World Motion Generation

For many algorithms, action trajectories generated in simulation is not align with those in the real world. We introduce two methods for generating trajectories in simulation that closely mimic real-world scenarios by leveraging prior knowledge of real-world manipulation trajectories.

Teleoperation. We've developed a lightweight, cost-effective teleoperation system requiring just one-click deployment. It facilitates simultaneous control of dexterous hands and grippers in both real-world and simulated settings (Figure 6). This system supports data collection for offline training, like diffusion policy.[67, 9]. Implementation details are in Appendix I

MoveIt. Incorporating MoveIt into our framework elevates motion planning and obstacle avoidance beyond heuristic trajectory methods, as noted in previous studies [57, 63]. Employing MoveIt for real-world robot execution also aids visual algorithms. Adapting models to MoveIt-generated trajectories during training reduces the sim-to-real gap. Detailed implementations are provided in Appendix H.

https://doi.org/10.52202/079017-0379

7 Experiments

7.1 Simulation Experiment Setup

Methods. We selected three vision-based and two reinforcement learning (RL) algorithms for experiment, with details listed in Table 4. For vision-based algorithms, we prioritized those utilizing dense representations for garments, as they have demonstrated generalization ability and are suitable for various downstream tasks.

Table 4: **Benchmark Methods of GarmentLab.**

Method	UniGarmentManip (UGM)[57]	DIFT[52]	Affordance[58]	RL-State[46]	RL-Vision[46]
Type	3D-Visual-Correspondence	2D-Visual-Correspondence	3D-Representation	RL	RL
BackBone	PointNet++[40]	Stable-Diffusion	PointNet++[40]	PPO	PPO
Input	Point Cloud	RGB Image	Point Cloud	State-Base GT	Partial Point Cloud

Tasks. Although we proposed many novel tasks, current algorithms cannot fully solve them. Thus, for large garments like tops, dresses, and trousers, we chose folding, hanging, and unfolding tasks. For small items like hats and gloves, we selected hanging and placing tasks to evaluate visual and RL algorithms. For dexterous and mobile tasks, existing work mostly employs RL algorithms. Hence, we evaluated the performance of both RL-state-based and RL-vision-based algorithms separately.

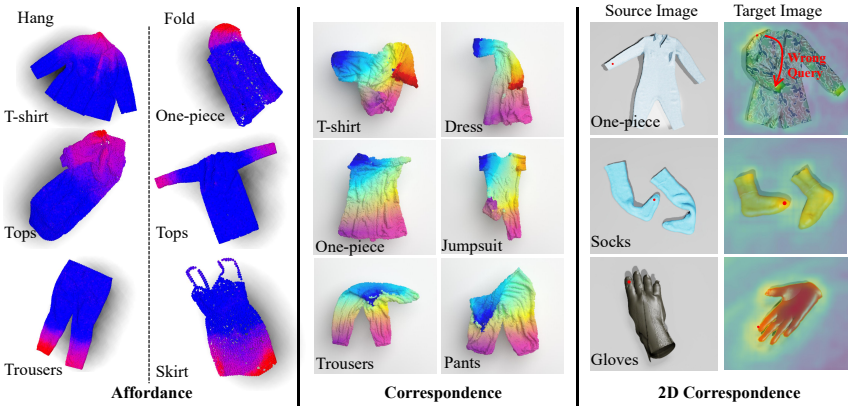


Figure 7: **Qualitative Results.** We visualize the qualitative results of the three vision-based algorithms: the left, middle, and right sections of this image correspond to the Affordance, Correspondence, and DIFT algorithms, respectively. Note that the DIFT exhibits query errors. For detailed analysis, please refer to Experiment Section.

7.2 Simulation Result and Analysis

Table 5 present quantitative comparisons between vision-based algorithms and RL-based algorithms. Figure 7 intuitively demonstrates the visual results of three different vision algorithms.

Table 5: **Simulation Results on Traditional Tasks.** Numbers in the Large-piece column represent scores for Tops, Trousers and Skirts. Numbers in the Small-piece column represent scores for Hats and Gloves.

	Large-piece			Small-piece	
Method	Fold	Unfold	Hang	Place	Hang
UGM	**61.5 / 62.1 / 59.8**	**58.3 / 60.5 / 57.2**	61.8 / 57.5 / 59.7	33.2 / 35.4	31.8 / 29.2
DIFT	32.7 / 36.7 / 31.2	18.7 / 23.3 / 17.6	31.2 / 27.6 / 29.7	**66.4 / 63.2**	**64.7 / 61.2**
Affordance	53.2 / 51.8 / 56.9	32.4 / 36.7 / 31.8	**64.1 / 60.2 / 61.3**	63.2 / 61.5	62.6 / 60.4
RL-State	14.8 / 12.5 / 9.8	6.5 / 8.8 / 12.7	13.1 / 19.7 / 14.7	14.2 / 12.1	12.8 / 13.2
RL-Vision	6.7 / 8.2 / 3.2	5.2/ 6.2/ 8.8	7.6 / 5.3 / 4.1	13.1 / 14.8	11.3 / 15.2

https://doi.org/10.52202/079017-0379

Vision-Based Algorithm. Among the three vision-based algorithms, UGM performed best on large-piece clothing, emphasizing cross-deform and cross-object consistency in learning representations, DIFT excels with small-piece clothing due to its robustness to object rotation but lacks proficiency in understanding clothing folding. Affordance works well for tasks that do not require precise point selection, such as hanging, but struggles with folded garments.

RL Algorithm. Compared to vision-based algorithms, RL performs poorly on garment manipulation due to the complex dynamics of garments. Our analysis of training videos showed that RL often generates abnormal trajectories, causing clothes to get tangled with the robotic arm or be pushed away. This issue is more pronounced with RL-vision-based methods, as the higher-dimensional and partial visual observations hinder the model's ability to converge on an effective strategy. For dexterous and mobile tasks, the larger action and search spaces result in suboptimal performance. Further discussion and analysis can be found in Appendix B.

7.3 Real-World Experiments

In our real-world experiments, we focused on testing vision-based algorithms due to the risk associated with RL actions. T-shirts for folding and hats for hanging, were selected for experimentation. Additionally, we conducted ablation study on proposed sim2real methods using UGM (Table 6). Our real-world results align with our simulation findings, indicating GarmentLab environment can enhance real-world applications. For sim2 real algorithm, without point cloud alignment and noise augmentation along with keypoint embedding alignment can improve representation smoothness and accuracy. Qualitative sim2real results are shown in Figure 6 (Right).

Table 6: **Real-World Experiment and Ablation Results**, w/o PA, w/o Noise, and w/o EA respectively represent UniGarmentManip without Pointcloud Alignment, Noise Augmentation, and KeyPoint Embedding Alignment.

Method	UGM	DIFT	Affordance	w/o PA	w/o Noise	w/o EA
Tshirt-Folding	**10**/15	8/15	6/15	2/15	8/15	7/15
Hat-Hanging	10/15	**14**/15	9/15	5/15	8/15	9/15

8 Conclusion

We introduce GarmentLab, a comprehensive environment and benchmark for manipulating garments and deformable objects. GarmentLab includes the GarmentLab Engine, supporting various simulation methods and ROS integration; GarmentLab Assets, a diverse dataset of robots, materials, and garments; and GarmentLab Benchmark, proposing several novel tasks. It also provides the first real-world deformable benchmark along with several sim2real methods.

9 Acknowledgment

This project is supported by The National Natural Science Foundation of China (No. 62376006), the National Youth Talent Support Program (8200800081) and the National Natural Science Foundation of China (No. 62136001).

References

[1] Yahav Avigal, Lars Berscheid, Tamim Asfour, Torsten Kröger, and Ken Goldberg. Speedfolding: Learning efficient bimanual folding of garments. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2022.

[2] Arpit Bahety, Shreeya Jain, Huy Ha, Nathalie Hager, Benjamin Burchfiel, Eric Cousineau, Siyuan Feng, and Shuran Song. Bag all you need: Learning a generalizable bagging strategy for heterogeneous objects. *IROS*, 2023.

[3] Jan Bender, Matthias Müller, and Miles Macklin. Position-based simulation methods in computer graphics. In *Eurographics*, 2015.

[4] Hugo Bertiche, Meysam Madadi, and Sergio Escalera. Cloth3d: clothed 3d humans. In *European Conference on Computer Vision*, pages 344–359. Springer, 2020.

[5] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 510–517, 2015.

[6] Alper Canberk, Cheng Chi, Huy Ha, Benjamin Burchfiel, Eric Cousineau, Siyuan Feng, and Shuran Song. Cloth funnels: Canonicalized-alignment for multi-purpose garment manipulation. In *International Conference of Robotics and Automation (ICRA)*, 2022.

[7] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, L. Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *ArXiv*, abs/1512.03012, 2015.

[8] Wei Chen, Dongmyoung Lee, Digby Chappell, and Nicolas Rojas. Learning to grasp clothing structural regions for garment manipulation tasks. *arXiv preprint arXiv:2306.14553*, 2023.

[9] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric A. Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *ArXiv*, abs/2303.04137, 2023.

[10] Sachin Chitta, Ioan Alexandru Sucan, and Steve B. Cousins. Moveit! [ros topics]. *IEEE Robotics Autom. Mag.*, 19:18–19, 2012.

[11] Philippe G. Ciarlet. The finite element method for elliptic problems. In *Classics in applied mathematics*, 2002.

[12] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. *arXiv*, 2016.

[13] Congyue Deng, Or Litany, Yueqi Duan, Adrien Poulenard, Andrea Tagliasacchi, and Leonidas J. Guibas. Vector neurons: A general framework for so(3)-equivariant networks. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12180–12189, 2021.

[14] Peter Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *Conference on Robot Learning*, 2018.

[15] Haoran Geng, Ziming Li, Yiran Geng, Jiayi Chen, Hao Dong, and He Wang. Partmanip: Learning cross-category generalizable part manipulation policy from point cloud observations. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2978–2988, 2023.

[16] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023.

[17] Huy Ha and Shuran Song. Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding. In *Conference on Robot Learning*, pages 24–33. PMLR, 2022.

 https://doi.org/10.52202/079017-0379

[18] Minho Heo, Youngwoon Lee, Doohyun Lee, and Joseph J. Lim. Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation. In *Robotics: Science and Systems*, 2023.

[19] Sebastian Hofer, Kostas E. Bekris, Ankur Handa, Juan Camilo Gamboa, Florian Golemo, Melissa Mozifian, Christopher G. Atkeson, Dieter Fox, Ken Goldberg, John Leonard, C. Karen Liu, Jan Peters, Shuran Song, Peter Welinder, and Martha White. Perspectives on sim2real transfer for robotics: A summary of the r: Ss 2020 workshop. *ArXiv*, abs/2012.03806, 2020.

[20] Sebastian Höfer, Kostas E. Bekris, Ankur Handa, Juan Camilo Gamboa, Melissa Mozifian, Florian Golemo, Christopher G. Atkeson, Dieter Fox, Ken Goldberg, John Leonard, C. Karen Liu, Jan Peters, Shuran Song, Peter Welinder, and Martha White. Sim2real in robotics and automation: Applications and challenges. *IEEE Trans Autom. Sci. Eng.*, 18:398–400, 2021.

[21] Yuanming Hu, Luke Anderson, Tzu-Mao Li, Qi Sun, Nathan A. Carr, Jonathan Ragan-Kelley, and Frédo Durand. Difftaichi: Differentiable programming for physical simulation. *ArXiv*, abs/1910.00935, 2019.

[22] Zhiao Huang, Yuanming Hu, Tao Du, Siyuan Zhou, Hao Su, Joshua B. Tenenbaum, and Chuang Gan. Plasticinelab: A soft-body manipulation benchmark with differentiable physics. In *ICLR*, 2021.

[23] Stephen James, Z. Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5:3019–3026, 2019.

[24] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross B. Girshick. Segment anything. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3992–4003, 2023.

[25] Cheng-I Lai. Contrastive predictive coding based feature for automatic speaker verification. *arXiv preprint arXiv:1904.01575*, 2019.

[26] Michael J. Landau, Benjamin Y. Choo, and Peter A. Beling. Simulating kinect infrared and depth images. *IEEE Transactions on Cybernetics*, 46(12):3018–3031, 2016.

[27] Anqi Li, Ching-An Cheng, Muhammad Asif Rana, Mandy Xie, Karl Van Wyk, Nathan D. Ratliff, and Byron Boots. Rmp2: A structured composable policy class for robot learning. *ArXiv*, abs/2103.05922, 2021.

[28] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023.

[29] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B. Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. In *International Conference on Learning Representations*, 2019.

[30] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 432–448. PMLR, 2021.

[31] Liu Liu, Wenqiang Xu, Haoyuan Fu, Sucheng Qian, Qiaojun Yu, Yang Han, and Cewu Lu. Akb-48: A real-world articulated object knowledge base. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14809–14818, 2022.

[32] Miles Macklin, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33:1 – 12, 2014.

[33] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu based physics simulation for robot learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[34] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023.

[35] Kaichun Mo, Shilin Zhu, Angel X. Chang, L. Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 909–918, 2018.

[36] Yoshihiko Nakamura and Hideo Hanafusa. Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems Measurement and Control-transactions of The Asme*, 108:163–171, 1986.

[37] Open Robotics. Franka emika panda robot. `https://robodk.com/robot/Franka/Emika-Panda`, June 2024.

[38] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2017.

[39] Xavi Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Tsung-Yen Yang, Ruslan Partsey, Ruta Desai, Alexander William Clegg, Michal Hlavac, So Yeon Min, Vladimr Vondru, Thophile Gervet, Vincent-Pierre Berges, John Turner, Oleksandr Maksymets, Zsolt Kira, Mrinal Kalakrishnan, Devendra Jitendra Malik, Singh Chaplot, Unnat Jain, Dhruv Batra, † AksharaRai, and † RoozbehMottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots. *ArXiv*, abs/2310.13724, 2023.

[40] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[41] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dietor Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *RSS*, 2023.

[42] Morgan Quigley. Ros: an open-source robot operating system. In *IEEE International Conference on Robotics and Automation*, 2009.

[43] Reallusion. Actorcore. `https://actorcore.reallusion.com/3d-motion`, 2024.

[44] Shadow Robot. Shadow dexterous hand series. `https://www.shadowrobot.com/dexterous-hand-series/`, 2024.

[45] Rockwell. Ridgeback. `https://clearpathrobotics.com/ridgeback-indoor-robot-platform/`, 2024.

[46] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[47] Daniel Seita, Pete Florence, Jonathan Tompson, Erwin Coumans, Vikas Sindhwani, Ken Goldberg, and Andy Zeng. Learning to Rearrange Deformable Cables, Fabrics, and Bags with Goal-Conditioned Transporter Networks. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[48] Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. *arXiv preprint arXiv:2306.14447*, 2023.

[49] Dave Shreiner, Mason Woo, Jackie Neider, and Tom Davis. Opengl programming guide: the official guide to learning opengl. In *Shreiner*, 1993.

[50] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B. Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se(3)-equivariant object representations for manipulation. *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400, 2021.

 https://doi.org/10.52202/079017-0379

[51] P. Slusallek, Peter Shirley, William R. Mark, Gordon Stoll, and Ingo Wald. Introduction to real-time ray tracing. *ACM SIGGRAPH 2005 Courses*, 2005.

[52] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. *ArXiv*, abs/2306.03881, 2023.

[53] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[54] Qianxu Wang, Haotong Zhang, Congyue Deng, Yang You, Hao Dong, Yixin Zhu, and Leonidas J. Guibas. Sparsedff: Sparse-view feature distillation for one-shot dexterous manipulation. *ArXiv*, abs/2310.16838, 2023.

[55] Yufei Wang, Zhanyi Sun, Zackory Erickson, and David Held. One policy to dress them all: Learning to dress people with diverse poses and garments. In *Robotics: Science and Systems (RSS)*, 2023.

[56] Ruihai Wu, Kai Cheng, Yan Zhao, Chuanruo Ning, Guanqi Zhan, and Hao Dong. Learning environment-aware affordance for 3d articulated object manipulation under occlusions. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[57] Ruihai Wu, Haoran Lu, Yiyan Wang, Yubo Wang, and Dong Hao. Unigarmentmanip: A unified framework for category-level garment manipulation via dense visual correspondence. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[58] Ruihai Wu, Chuanruo Ning, and Hao Dong. Learning foresightful dense visual affordance for deformable object manipulation. In *IEEE International Conference on Computer Vision (ICCV)*, 2023.

[59] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas Guibas, and Hao Dong. VAT-mart: Learning visual action trajectory proposals for manipulating 3d ARTiculated objects. In *International Conference on Learning Representations*, 2022.

[60] F. Xia, Amir Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9068–9079, 2018.

[61] Zhou Xian, Bo Zhu, Zhenjia Xu, Hsiao-Yu Tung, Antonio Torralba, Katerina Fragkiadaki, and Chuang Gan. Fluidlab: A differentiable environment for benchmarking complex fluid manipulation. In *The Eleventh International Conference on Learning Representations*, 2023.

[62] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. Sapien: A simulated part-based interactive environment. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11094–11104, 2020.

[63] Han Xue, Yutong Li, Wenqiang Xu, Huanyu Li, Dongzhe Zheng, and Cewu Lu. Unifolding: Towards sample-efficient, scalable, and generalizable robotic garment folding. *ArXiv*, abs/2311.01267, 2023.

[64] Han Xue, Wenqiang Xu, Jieyi Zhang, Tutian Tang, Yutong Li, Wenxin Du, Ruolin Ye, and Cewu Lu. Garmenttracking: Category-level garment pose tracking. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21233–21242, 2023.

[65] Han Xue, Wenqiang Xu, Jieyi Zhang, Tutian Tang, Yutong Li, Wenxin Du, Ruolin Ye, and Cewu Lu. Garmenttracking: Category-level garment pose tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21233–21242, 2023.

[66] Kevin Zakka, Laura M. Smith, Nimrod Gileadi, Taylor A. Howell, Xue Bin Peng, Sumeet Singh, Yuval Tassa, Peter R. Florence, Andy Zeng, and P. Abbeel. Robopianist: A benchmark for high-dimensional robot control. *ArXiv*, abs/2304.04150, 2023.

[67] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy. *ArXiv*, abs/2403.03954, 2024.

[68] Fan Zhang and Yiannis Demiris. Learning grasping points for garment manipulation in robot-assisted dressing. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9114–9120. IEEE, 2020.

[69] Yan Zhao, Ruihai Wu, Zhehuan Chen, Yourong Zhang, Qingnan Fan, Kaichun Mo, and Hao Dong. Dualafford: Learning collaborative visual affordance for dual-gripper object manipulation. *International Conference on Learning Representations (ICLR)*, 2023.

[70] Bingyang Zhou, Haoyu Zhou, Tianhai Liang, Qiaojun Yu, Siheng Zhao, Yuwei Zeng, Jun Lv, Siyuan Luo, Qiancai Wang, Xinyuan Yu, Haonan Chen, Cewu Lu, and Lin Shao. Clothesnet: An information-rich 3d garment model repository with simulated clothes environment. *ICCV*, 2023.

[71] Zhehua Zhou, Jiayang Song, Xuan Xie, Zhan Shu, Lei Ma, Dikai Liu, Jianxiong Yin, and Simon See. Towards building ai-cps with nvidia isaac sim: An industrial benchmark and case study for robotics manipulation. *arXiv preprint arXiv:2308.00055*, 2023.

[72] Berk Çalli, Aaron Walsman, Arjun Singh, Siddhartha S. Srinivasa, P. Abbeel, and Aaron M. Dollar. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. *ArXiv*, abs/1502.03143, 2015.

 https://doi.org/10.52202/079017-0379

Appendix Overview

A GarmentLab Asset

GarmentLab asset contains a vast range of objects which are commonly seen in our daily life, from rigid object, articulated object, garment with various materials to all kinds of robots and human models.

B Experiment

Details about how the experiments described in the main paper is conducted, such as partition of training and testing sets and evaluation metrics definition.

C Related Work

The main reference work under the topic of traditional embodied and robotic simulator, deformable objects and cloth Benchmark, downstream tasks and algorithms of garment manipulation.

D Physics Simulation

Details about the simulation methodology of different types of physics simulation, from garment, deformable body, rigid body to wind and fluid. Discussion of how the changing of physics parameters leads to different simulation performance.

E Sim2Real

More detailed description of sim-to-real visual alignment and motion generation methods with specific algorithm process and quantitative mathematical formulas.

F Real-World Benchmark

Discussion about the principle and consideration of object selection. Detailed process of scanning real-world objects into simulation models and protocol benchmark guidelines of garment manipulation.

G Task

The task categories and detailed settings with multi-physics interactions, including but not limited to garment-garment task, garment-avatar task. Long-horizon tasks are also discussed here.

H MoveIt

How trajectories generated by MoveIt are recorded and how to adapt visual models in the simulator to the trajectories, aiming to narrow sim-to-real gap.

I Teleoperation

Detailed process of how the motion of human demonstration is captured and then adapted to embodiment, bringing robots with action in human-level intelligence.

J Limitation

Limitation of our work.

K Broader Impact

The potential societal impacts of our work.

A GarmentLab Assets

- **Rigid Object** We mainly import objects from ShapeNet[7],PartNet[35]and YCB dataset[72]. Note that we have filtered out objects that are not suitable for physical simulation and have issues interacting with garments or fluid, and then reorganized and reclassified the dataset.
- **Articulated Object** Having much higher degree-of-freedom(DoF) state spaces, articulated objects are, however, generally more difficult to understand and interact with compared to 3d rigid objects. We mainly import articulated objects from PartNet-Mobility dataset [62] including Chair, Box, Bucket, Washing machine and Storage Furniture etc, to establish the comprehensive tasks for indoor robots such as folding clothes and putting them into the wardrobe.
- **Garment and Cloth** We select garments from ClothesNet [70], a large-scale dataset of 3D clothes objects with information-rich annotations. We select garments from 11 categories including Hat, Tie, Mask, Gloves and Socks and use two physical simulation method to simulate them. We also include standard square cloths, like dishcloths and tablecloths, to cover indoor task needs. Note that there are still gaps between meshes and ready-to-simulate object, we do post-processing of garments including giving correct physical parameters to simulate them.
- **Robot**. We deploy a variety of specialized robots for diverse tasks, including a 7-DoF Franka manipulator[37] with a parallel gripper, a UR5 with suction for manipulation tasks, and a RidgebackFranka[45] with wheels for mobility and navigation. For dexterous tasks, we use ShadowHands[44] mounted on a UR10e.
- **Human Model**. We incorporate human model to construct long-horizon tasks, such as dressing up. Utilizing avatars selected from actorcore[43], we assign specific motions to each avatar to facilitate collaboration with robots. Each avatar comprises articulated joints, surface skin mesh, and clothing, enabling realistic simulation of human structure and motion.
- **Materials**. Materials are crucial components of virtual relightable assets, defining the interaction of light at the surface of geometries. We carefully choose materials from the Omniverse Base Material library to attain optimal rendering outcomes, a critical aspect for visual-based algorithms. Moreover, diverse textures can aid algorithms in understanding the relationship between an object's appearance and its physical behavior.

B Experiment

B.1 Overview

Generalization ability. As a novel environment, **GarmentLab** especially focus on evaluating and improving the generalization abilities of algorithm. We evaluate the generalization ability from the following aspects. **Novel Object** Thanks to rich GarmentLab Asset, we split garment and other object dataset into *Train/Var/Test* at proportion 70%/15%/15% to test algorithms generalization ability on object level. Moreover, as garment and deformable object have nearly infinite self-deform state, we introduce **Novel State**. For example, we disturb garment initial state and test model's ability on handling wrinkled and folded clothes. Moreover, in order to improve algorithm ability of planning and collision avoidance, we also involve **Novel Scene**, as for task like make up table, the shadow of irrelevant object can also influence navigation.

Metrics. We primarily use the success rate as the evaluation metric. It is important to note that because garments and fluids can easily change state due to gravity or friction, we consider a task successful if it meets the success criteria and maintains this state for at least five seconds. For tasks that appeared in previous work, we adopted the widely accepted success criteria and tolerance thresholds from the goal state. For example, we use Intersection-over-Union (IOU) between the target and the folded garments to evaluate folding task[57, 6] and use coverage area to evaluate unfolding task[17]. For novel tasks such as washing or blowing, the goal states and tolerances are derived according to human behaviors.

https://doi.org/10.52202/079017-0379

B.2 Experiment Task Setting

For large garments like tops, dresses, and trousers, we chose folding, hanging, and unfolding tasks. And for small items like hats and gloves, we selected hanging and placing tasks to evaluate visual and RL algorithms. The detailed experiment settings for the tasks listed above are shown below:

Garment-hanging task requires robots to hang garments to a fixed couple. The first criterion of success is that the garment can be hanged steadily (five seconds in real experiment) on the couple. Then to ensure that the garment is hanged in the right pose (not hanged at sleeve or other strange cases), we compare the manipulation result with a standard human demonstration by computing the sum particle-to-particle distance between two states. The total distance within a predefined value will be regarded as success. The initial states of garment to hang is obtained by dropping from random initial poses over the ground.

Unfolding task requires robots to unfold garments at random deformations to be flat. Follow ClothFunnel[6] unfolding task success when the garment ground-truth vertices are within a reasonable range of the initial state, i.e., the flat state before the vertices were disrupted. This is because we found that using coverage area as defined by FlingBot[17] may not reasonably reflect success as the garment structure becomes more complex. The initial states of garment to unfold is obtained by dropping from random initial poses over the ground.

Folding task requires robots to fold garments from a flat states. After manipulation, we calculate the particle-to-particle distance between a.the final state of garment after manipulation trial and b.the garment state obtained by human demonstration. The manipulation whose total distance is lower than a predefined value can be regarded as success. The initial states of garments to fold are obtained by placing flatly on random position with small disturbance.

Hat-Hanging task requires robots to hang hats to a fixed couple. The criterion of success is that the hat can be hanged steadily on the couple and would not fall on the ground. The initial states of hat to hang is obtained by dropping from random initial poses over the ground.

Placing task requires robots to get the hats/gloves which are previous hanged at the couple. The placing task succeeds when the hats/gloves are fetched and placed on the right position without falling on the ground. The initial states of hats/gloves to fetch and place are obtained by random dropping from a random poses over the couple, where only the successfully hanged cases will be used for training and testing.

B.3 Detailed Analysis

Vision-Based Algorithm. Comparing the three vision models, we found that UniGarmentManip (UGM) have the best performance on Large-piece of garments. We conjecture that this stems from (1) The consistence of representation on self-deformations. As model have the understanding of deformation on garments, it is easier to detect keypoints required in Folding tasks and Fling tasks in diverse garment states. (2) The explicit design of cross-object representation consistency. UniGarmentManip use skeleton(a graph of keypoints) as the shared bridge for different garments with similar structures, which makes model have the ability to understand the topology of the 3D object in the same category. However, we found that this result is not as good as the original paper using PyFlex. This could be because we introduced the robot and the scene here, so some invalid selection points, such as those beyond the robot's reach or causing collisions, were considered unsuccessful. Additionally, while the original paper used only T-shirts, our study included jackets and other garments with front openings, this wider variety of clothes also increased the complexity of our task. For Affordance, we found that it performs well in Hanging tasks possibly because of its task-specific designed which makes it chooses the grasp points more accurately. In contrast, DIFT has poor performance on these three tasks especially on unfolding tasks due to the unawareness of garment deformations on 2D pretrained correspondence. This is reasonable because most objects in world for training do not have garment-level deformations. However, DIFT perfors better with small-piece garments like hats and gloves due to their minimal deformation, Besides, the pretrain model based on large diffusion model are more robust to rotation, which is crucial for handling small clothes. For UniGarmentManip and Affordance, they are not 3D-equivariant models, so they are more sensitive to rotations, resulting in poorer performance with small clothes compared to DIFT.

RL-Based Algorithm. We modified the traditional PPO by directly replacing the value net of PPO with the success information from the simulation ground-truth information. This is because in robotics tasks, the value of the policy can be easily obtained from the ground-truth information.At the same time, following ClothFunnel[6] and FlingBot[17], we primarily used RL for selecting points and adopted a scripted policy for the trajectory, with simple adaptations based on the selected points. This is because directly using RL to train the trajectory is particularly unstable. We will elaborate on this point in more detail below.

We found that PPO's performance on state-based tasks was significantly worse than visual algorithms. After analyzing the training videos, we identified several reasons for this: (1) Abnormal trajectories of the robotic arm caused collisions and pushed the clothes far away. (2) Large reward fluctuations in long-horizon tasks led to training instability, as the robotic arm's random folding actions in the early stages caused significant reward variations. (3) Completing long-horizon tasks was difficult due to the robotic arm's abnormal trajectories disrupting previous steps, such as interfering with the sleeves during the folding task. (4) Wide-ranging movements of the robotic arm caused clothes to wrap around it, particularly in the hanging task, leading to failure. The visual algorithm avoided these issues because its execution trajectories were mostly predefined, such as pick-and-place or fling trajectories.

We also found that the performance of visual-based PPO is significantly inferior to state-based PPO, due to the following reasons: (1) The higher dimensionality of visual input makes training more difficult. (2) The visual input consists of partial point clouds, which can confuse the model, especially for thin objects like clothing. (3) Detecting the object position is more challenging for visual input, resulting in algorithmic failures during the grasping stage. These findings are consistent with those of SoftGym [30].

B.4 Training Details of Main Algorithms

B.4.1 UniGarmentManipulation (UGM)

For **Hyper-parameters selection**, we set batch size to be 32. In each batch, we sample 32 garment pairs. For each garment pair, we sample 20 positive and 150 negative point pairs for each positive point pair. Therefore, in each batch, $32 \times 32 \times 20$ data will be used to update the model. During the Correspondence training stage, we train the model for 40,000 batches. During Coarse-to-fine Refinement, we train the model for 100 batches. During Few-shot Adaptation, we slightly refine the model using 5 demonstration data. Besides, we set the number of skeleton pairs to be 50.

For **computational resource**, we use PyTorch as our Deep Learning framework. Each experiment is conducted on an RTX 3090 GPU, and consumes about 22 GB GPU Memory for training. It takes about 12 hours to train the Coarse Stage, with 1-2 hours of Coarseto-fine Refinement and 0.5 hour's Few-shot Adaptation.

B.4.2 Affordance

For **Hyper-parameters selection**, we set batch size to be 128, where each pair contains one positive manipulation point and one negative manipulation point on the same garment, automatically balancing the training data. "Positive" means manipulating on that point can lead to the success of the whole task while "negative" means failure. In each batch, 128×2 data will be used to update the model. During the Affordance training stage, we train the model for 36,000 batches. The model is designed to make binary classification with cross-entropy as loss function. The output of affordance model reflects the success rate when manipulating on that point, which ranges from 0 to 1. During manipulation, we just select the point with the highest score to manipulate.

For **computational resource**, we use PyTorch as our Deep Learning framework. Each experiment is conducted on an RTX 3090 GPU, and consumes about 16 GB GPU Memory for training. It takes about 18 hours to train the model for a task with a specific category of garment.

B.4.3 DIFT

As pretrained model, DIFT use stable-diffusion as backbone. For **Hyper-parameters selection and prompt enginering** We use the default parameters of DIFT. We crop the image size to 762×762 and set timestep for diffusion to 261. The ensemble size was set to 8. We use official network architecture

 https://doi.org/10.52202/079017-0379

pipeline followed by our own designed robot excution pipeline. The robot execution pipeline is similar to UniGarmentManip. We only substitute the query model to DIFT. For **computational resource**, we use PyTorch as our Deep Learning framework. Each experiment is conducted on an RTX 3090 GPU, and consumes about 20 GB GPU Memory for inferencing.

C Related Work

Traditional Embodied and Robotic Simulator The simulator plays an indispensable role in robotics development as it allows for the rapid and safe acquisition of vast amounts of interaction data, facilitating the implementation of various algorithms. However, the majority of mainstream robot simulators[53, 12, 33, 62] primarily support rigid object simulation including the collision and friction between them. Besides, most of robot simulators are CPU-based[39, 66, 64], severely limiting their parallel capabilities and resulting in slow training speeds. Additionally, these simulations exhibit a significant sim2real gap due to the absence of comprehensive sim2real algorithm designs[20, 38, 19].Nevertheless, based on Isaac Sim[71], our benchmark not only supports parallel data collection but also incorporates comprehensive sim2real designs, including RL-based and vision-based algorithms.
Deformable and Cloth Benchmark In recent years, there has been a surge in deformable and garment simulation environments[30, 22, 61]. However, the most server problem of these kinds of simulation or benchmarks is that they can only simulate certain kinds of objects as they only support one simulation method, which makes it impossible to explore the physical interaction between multiple kinds of objects. Moreover, these benchmarks are lack of diversity as they are built directly on the underlying simulation architecture and have not integrated with mature platforms, thereby limiting the range of simulated objects and scenes. For instance, softgym[30], built on NVIDIA Flex[32], is confined to simulating tops and trousers while fluidlab[61], built on Taichi[21], can only simulate fluid and performs poorly on rigid objects simulation. Additionally, many benchmarks[61, 30] lack the ability to import robots and establish real grasps, posing significant challenges for joint control and vision-based algorithms. By contrast, GarmentLab provides sophisticated 3D meshes and facilitates various simulation techniques, enabling the modeling of garments, fluids, flow dynamics, avatars, rigid and articulated objects, and their interactions. This inclusive and adaptable platform offers a more comprehensive solution for research and development. The full detailed comparison of out benchmark between others can be found in Appendix A.
Garment and cloth manipulation Manipulating a single garment or cloth is a well-studied area, with previous works focusing on learning policies for specific tasks such as folding [1, 63], unfolding [17], grasping [8, 68], and dressing-up [55]. However, as many daily tasks involve interactions between various physical media, current algorithms often fall short in solving real-life tasks. Although many proposed algorithms have full potential to solve these problems[59, 29], they are hindered by the lack of a mature simulation platform capable of supporting such simulations. Furthermore, while current research predominantly emphasizes gripper manipulation tasks, we introduce tasks utilizing suction, dexterous hands, and mobile robots. We believe that GarmentLab will make a unique and valuable contribution to the robotics community by providing a new platform for developing garment manipulation algorithms and significantly expanding the scope of existing methods.

D Physics Simulation

D.1 Modeling methodology

D.1.1 Position-Based Dynamics (PBD) for Garment

Position-Based Dynamics (PBD) is an efficient and stable method for simulating cloth, particularly suitable for complex garments like dresses. PBD models deformable objects as systems of interconnected particles governed by constraints that dictate their physical interactions and behaviors. In PBD, a dress is represented as a triangular mesh where particles serve as discrete points on the cloth surface with attributes such as position x_i, velocity v_i, and inverse mass w_i. The method operates by directly manipulating particle positions to satisfy a series of constraints, achieving stable simulations of deformable materials. These constraints include stretching constraints, which enforce distance maintenance between neighboring particles to prevent excessive elongation, mathematically defined as $C(x_i, x_j) = \|x_i - x_j\| - d$, where d is the rest distance between particles x_i and x_j. Bending constraints maintain angles between adjacent triangles in the mesh to simulate resistance to

bending, formulated as $C(x_i, x_j, x_k)$, where the constraint function depends on the angle between particle triplets. Collision constraints detect and resolve collisions between particles and other objects, ensuring realistic interactions within the environment. The PBD algorithm involves initializing particles and constraints based on the dress's geometry and material properties, applying external forces such as gravity, predicting new particle positions as $\hat{p}_i = p_i + \Delta t \cdot v_i$ where Δt is the time step, iteratively adjusting particle positions to satisfy constraints, updating particle velocities, and determining final positions for each time step.

D.1.2 Position-Based Dynamics (PBD) for Fluid Simulation

Position-Based Dynamics (PBD) is a powerful method for simulating fluids due to its computational efficiency and stability. PBD treats fluids as collections of particles, where each particle represents a small volume of the fluid. Constraints are applied to ensure physical properties such as incompressibility and realistic fluid behavior. In fluid simulation, particles are characterized by attributes such as position x_i, velocity v_i, and inverse mass w_i.

A key constraint type in fluid simulation is the density constraint, which ensures that the fluid maintains a constant density. The density constraint for a particle i can be defined as:

$$C_i(\mathbf{x}) = \left(\sum_j m_j W(\|x_i - x_j\|, h) \right) - \rho_0$$

where W is the smoothing kernel function, h is the smoothing length, m_j is the mass of particle j, and ρ_0 is the rest density of the fluid. Collision constraints handle interactions between fluid particles and solid boundaries, ensuring particles do not penetrate solid objects.

The PBD algorithm steps for fluid simulation include initializing fluid particles with positions, velocities, and masses, applying external forces such as gravity, computing predicted positions $\hat{p}_i = p_i + \Delta t \cdot v_i$, adjusting particle positions to satisfy density and collision constraints, updating particle velocities based on the corrected positions, and integrating the updated positions and velocities for the current time step.

D.1.3 Finite Element Method (FEM) for Simulating Deformable Objects

The Finite Element Method (FEM) is a robust numerical technique for simulating the mechanical behavior of deformable objects, ideal for intricate geometries and diverse material properties, such as a toy bear. FEM discretizes the object into a mesh of finite elements and solves the equations of motion to accurately capture realistic deformations under various forces.

In FEM, the deformable object is represented by a mesh consisting of nodes and elements. Nodes are points where the equations of motion are solved, and elements are polyhedral shapes, such as tetrahedrons, that connect these nodes. Material properties, including elasticity, density, and damping, determine the response of the object to applied forces.Modeling a deformable body involves several key steps. First, a deformable body component is added to the mesh, which generates collision and simulation tetrahedral (tet) meshes from the source mesh. The mesh is then separated into visualization, collision, and simulation tetmeshes, each serving distinct purposes in rendering, collision resolution, and simulation. Configuring the material properties involves defining characteristics such as stiffness and dynamic friction by creating and binding a new deformable body material.

D.1.4 Flow Models for Simulating Wind Effects

Flow models are essential for simulating wind effects, capturing the interactions between fluid (air) and objects. These models represent phenomena such as airflow, turbulence, and aerodynamic forces. The typotypoequations form the core of flow models and include the continuity equation for mass conservation $\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$, where ρ is the fluid density and $\mathbf{u}$ is the velocity vector; the momentum equation for force balance $\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = -\nabla p + \nabla \cdot \tau + \rho \mathbf{g}$, where p is the pressure, τ is the stress tensor, and $\mathbf{g}$ is the gravitational acceleration; and the energy equation for thermal effects $\frac{\partial (\rho E)}{\partial t} + \nabla \cdot (\rho E \mathbf{u}) = -\nabla \cdot \mathbf{q} + \tau : \nabla \mathbf{u} + \rho (\mathbf{g} \cdot \mathbf{u})$, where E is the total energy per unit mass and $\mathbf{q}$ is the heat flux vector.

https://doi.org/10.52202/079017-0379

Wind effects are modeled by defining wind sources, preparing high-resolution meshes, setting boundary conditions, configuring the appropriate flow model (e.g., LES or RANS), and running the simulation to compute wind interactions iteratively. This approach ensures accurate and dynamic representations of wind effects in various environments.

D.1.5 Rigid Body Simulation

Rigid body models are essential for simulating solid objects that move and interact based on physical laws without deforming. These simulations accurately represent the dynamics of solid objects under various forces. Key components include a rigid body component, which provides properties like linear and angular velocity, and a collision component, which defines how the body collides with other objects.The dynamics of rigid bodies are governed by solvers such as Temporal Gauss-Seidel (TGS) and Projected Gauss-Seidel (PGS), which ensure stability and efficiency. TGS improves convergence by considering temporal aspects of the simulation, while PGS iteratively projects velocities to satisfy constraints.Rigid bodies interact through collisions defined by collision shapes, which can be approximated using convex hulls, bounding shapes, or signed distance fields (SDFs). These approximations balance accuracy and computational performance.Mass properties of rigid bodies are derived from the volume and density of their collision geometries. For more precise control, explicit mass or density values can be set using a Mass component. This allows for accurate simulation of complex interactions and dynamic behaviors.

D.2 Multi-Physics Simulation Parameters Table

To maximize the value of different simulation methods, we assigned different parameters to various objects. In Table 7, we list all the adjustable parameters.

D.3 Parameter Effects on Physical Properties

In most cases, changes in parameters do not significantly alter the physical properties. For PBD simulations involving garment, the Particle Contact Offset parameter affects the thickness of the fabric; as its value increases, the fabric becomes progressively thicker. The Rest Offset parameter influences the distance between the dress and the ground upon landing, with an increase in this value resulting in a greater distance between the dress and the ground after it lands.

For PBD simulations involving fluid, the Velocity parameter affects the flow rate of the liquid; as its value increases, the liquid flows faster. The Cohesion parameter affects both the shape and flow rate of the liquid; at lower values, the liquid falls quickly and splashes out. As the value increases, the liquid flow slows down and splashing decreases, eventually leading to a smooth flow. The Particle Contact Offset parameter affects the form of the liquid as it falls; as its value increases, the liquid transitions from a continuous stream to a segmented, chunk-like flow.

In simulations involving deformable bodies, the Vertex Velocity Damping parameter affects the fall speed of objects such as hats; as the value increases, the fall speed decreases gradually. The Settling Threshold parameter also influences the fall speed of hats; increasing its value results in a slower fall speed, but once the value exceeds 1, the fall speed stabilizes. The Elasticity Damping parameter impacts the shape of the hat; as the value increases, the hat gradually collapses from a firm structure to a flat plane. The Young's Modulus parameter also affects the shape of the hat; at lower values (around 1e3), the hat collapses into a smaller height. As the value increases, the hat becomes firmer, and when the value reaches around 1e4, the hat initially stays firm and then gradually collapses. At a value of 15000, the hat remains completely firm.

For rigid body simulations, the Max Linear Velocity parameter affects the fall speed of rigid bodies such as hats; as the value increases, the fall speed decreases. When the value exceeds 50, the object practically stops falling.

In the context of flow simulations, the X-Component, Y-Component, and Z-Component parameters together determine the direction of the wind vector, while the Magnitude parameter determines the strength of the wind.

Table 7: Multi-physics parameters

Type	Parameters	Function	Range
Garment	Particle Contact Offset	Distance at which particles start interacting	0.03 - 0.12
	Contact Offset	Distance at which collisions are detected	0 - 16384
	Rest Offset	Distance at which particles are in resting contact	0 - 0.05
	Solid Rest Offset	Distance for particle-solid interactions	Default
	Fluid Rest Offset	Distance for particle-fluid interactions	Default
	Solver Position Iteration Count	Number of iterations for solver to satisfy constraints	6 - 255
	Max Depenetration Velocity	Maximum speed at which particles are separated when overlapping	inf
	Max Neighborhood	Maximum number of neighboring particles for interactions	36 - 512
	Density	Mass per unit volume of the material	default
	Friction	Resistance to sliding motion	default
	Damping	Reduction of motion or oscillations	default
	Viscosity	Internal friction within the fluid material	default
	Cohesion	Attractive force between particles	default
	Surface Tension	Elastic tendency of the material's surface	default
	Drag	Resistance experienced when moving through fluid or air	default
	Lift	Force acting perpendicular to fluid flow around the material	default
Fluid	Particle Contact Offset	Distance at which particles start interacting	0.17 - 0.3
	Contact Offset	Distance at which collisions are detected	default
	Rest Offset	Distance at which particles are in resting contact	0.03 - 0.2
	Solid Rest Offset	Distance for particle-solid interactions	0.1-0.2
	Fluid Rest Offset	Distance for particle-fluid interactions	0.1-0.15
	Solver Position Iteration Count	Number of iterations for solver to satisfy constraints	6 - 255
	Max Depenetration Velocity	Maximum speed at which particles are separated when overlapping	inf
	Max Neighborhood	Maximum number of neighboring particles for interactions	36 - 512
	Density	Mass per unit volume of the material	0 - 1e10
	Friction	Resistance to sliding motion	0 - 0.2
	Damping	Reduction of motion or oscillations	0 - 10
	Viscosity	Internal friction within the fluid material	1e3 - 1e6
	Cohesion	Attractive force between particles	0 - 100
	Surface Tension	Elastic tendency of the material's surface	0 - 100
	Drag	Resistance experienced when moving through fluid or air	0 - 78
	Lift	Force acting perpendicular to fluid flow around the material	0 - 1e10
Deformable Body	Vertex Velocity Damping	Rate of reduction of vertex velocities	0 - 10
	Simulation Mesh Resolution	Granularity of the simulation mesh	10
	Solver Position Iterations	Number of iterations for solver to satisfy positional constraints	8 - 255
	Sleep Threshold	Velocity below which the body is considered to be at rest	0 - 1e7
	Settling Threshold	Velocity below which the body is considered to have settled	0 - 1e7
	Sleep Damping	Additional damping as the body approaches the sleep threshold	0 - 1e7
	Contact Offset	Distance at which collisions are detected	-inf
	Rest Offset	Distance at which particles are in resting contact	-inf
	Self Collision Filter Distance	Minimum distance to avoid self-collision	-inf
	Remeshing Resolution	Resolution for remeshing the input mesh	Default
	Target Triangle Count	Target resolution for quadric simplification	Default
	Max Depenetration Velocity	Maximum speed at which vertices can be separated when overlapping	inf
	Density	Mass per unit volume	Default
	Dynamic Friction	Resistance to sliding motion	0 - 2048
	Young's Modulus	Stiffness of the material	1e3 - 1e10
	Poisson's Ratio	Ratio of transverse to axial strain	0 - 0.499
	Elasticity Damping	Reduction of oscillations and vibrations	0 - 0.05
	Damping Scale	Adjusts the overall damping effect	0 - 1.0
Rigid Body	Max Linear Velocity	The rate of change of position of the rigid body.	0-50
	Max Angular Velocity	The rate of change of rotation of the rigid body.	0-1e10
	Collision Shape	Defines the shape used for collision detection.	default
	Contact Offset	Distance from the surface where collisions are detected.	-inf-inf
	Rest Offset	Effective contact distance from the surface.	-inf-inf
	Convex Hull	Approximation method for collision shape.	0-64
	SDF (Signed Distance Field)	Approximation method using signed distance field.	default
	Mass	Defines the mass of the rigid body.	0 to Inf
	Density	Defines the density of the rigid body material.	0 to 1000
	Friction	Resistance to sliding motion.	0 to 1
	Restitution (Bounciness)	Degree of elasticity of collisions.	0 to 1
	Material Density	Density of the material applied to the rigid body.	0 to 1000
Flow	X-Component	Flow rate in the x-direction	-inf-inf
	Y-Component	Flow rate in the y-direction	-inf-inf
	Z-Component	Flow rate in the z-direction	-inf-inf
	Magnitude	Overall magnitude of the flow	0-inf

E Sim2Real

Transferring models trained in simulator to reality is challenging and become a critical issue for robotic research.However, most Sim2Real techniques are not yet fully automated and require careful human oversight. **In this work, we present three visual sim2real methods which are fully automated and self-supervised**. We mainly conduct experiment follow [57], learning dense visual representation for garment before and after our alignment method. The results are shown in Figure 6

 https://doi.org/10.52202/079017-0379

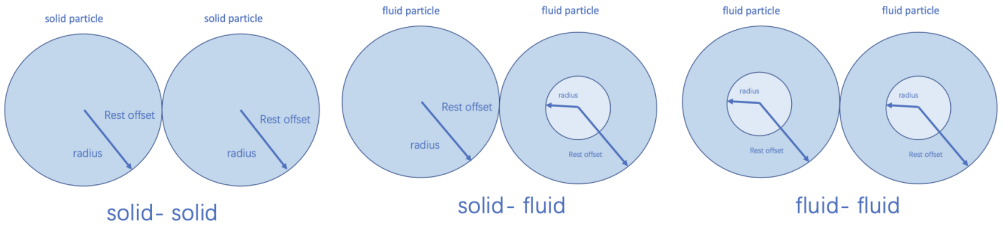


Figure 8: Different Types of Particle-Particle Interaction

E.1 Sim-Real Vision Alignment

Noisy Observation. Although previous do dedicated exploration on how to add noise to point cloud[26, 62], they need capture IR picture and do many calculation which is time-consuming. However, we have found that simply adding salt-and-pepper noise and Gaussian noise can already yield very good results. We directly add noise to depth picture and generate noised point cloud in the training data.

$$D_{\text{noised}}(x,y) = \begin{cases} D(x,y) + \mathcal{N}(0,\sigma^2) & \text{with probability } p_{\text{gaussian}} \\ D(x,y) + \text{salt} & \text{with probability } p_{\text{salt}} \\ D(x,y) - \text{pepper} & \text{with probability } p_{\text{pepper}} \\ D(x,y) & \text{otherwise} \end{cases}$$

where $D(x,y)$ represents the depth value at pixel (x,y) in the original depth picture, $\mathcal{N}(0,\sigma^2)$ represents Gaussian noise with mean 0 and variance σ^2, and salt and pepper noise is added with probabilities p_{gaussian}, p_{salt}, and p_{pepper} respectively.

For experiment, we add noise to the training data during the training process of dense visual correspondence[57]. As shown in figure6, before we do data argumentation for training data, the query results show many spots, indicating errors. This is due to discontinuities of dense representation caused by differences between the real-world point cloud and the simulator. After we add noise, the model become more robust to noise thus the representation become more smooth and accurate.

Point Cloud Alignment. Dense object descriptors [14] that learn point- or pixel-level object representations are proposed by and for robotic manipulation. The key idea of these works[57, 54, 50]is to represent an object as a function f that maps a 3D coordinate x to a spatial descriptor $z = f(x)$ of that 3D coordinate:$f(x) : \mathbb{R}^3 \to \mathbb{R}^n$. f may further be conditioned on point cloud $\mathbf{P} \in \mathbb{R}^{3\times N}$ and usually parameterized by a neural network. However, f are not always SE(3)-equivariant, which means to a rigid transform $(\mathbf{R},\mathbf{t}) \in SE(3)$, we can **NOT** guarantee that $f(\mathbf{x}|\mathbf{P}) \equiv f(\mathbf{Rx}+\mathbf{t}|\mathbf{RP}+\mathbf{t})$. However, in real world experiment, as the height and angle of the camera may be different from that in the simulator, the distributions of point cloud collected in simulation and real world are different. This will lead to wrong query result especially for garment as it highly rely on thickness to detect folding relationships.

Although we can choose SO(3)-equivariant network[13], the training of it is hard and time-consuming. Thus, we propose a direct way to align the point cloud in simulation and the realworld. As all rigid transform $(\mathbf{R},\mathbf{t}) \in SE(3)$ can be represented by affine matrix, we directly use gradient descent to optimize the affine matrix so that we can align the distribution between realworld point cloud and simulation point cloud. We chose the chamfer distance as the loss function because it is both robust to the various deform and shape of the garment and effectively aligns the positions. Equation 1 show our optimization objective and 2 show our loss function.

$$\text{Transforms}(R,t) = \begin{pmatrix} a & b & c & x \\ d & e & f & y \\ g & h & i & z \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

https://doi.org/10.52202/079017-0379

$$\text{Chamfer\_Loss}(A, B) = \frac{1}{|A|} \sum_{a \in A} \min_{b \in B} \|a - b\|^2 + \frac{1}{|B|} \sum_{b \in B} \min_{a \in A} \|a - b\|^2 \tag{2}$$

As shown in figure 6, before we align the point cloud, the model predict wrong result even in the flat case. After alignment, the model successfully predict the correct result.

Keypoint Embedding alignment. As the model learn point level representation, a direct way is to align the representation of corresponding point between realworld garment and simulation. In this part, we first attach marker to garment on the skeleton points(shoulder, end of the sleeve and bottom corner etc.) Then, we do self-play which enable franka to randomly choose the pick-and-place point to create garment deformation status. Then we use SAM[24] to detect key point and align correspondence key point with the simulation result. After we get ground-truth key point pair between simulation and realworld,we employ InfoNCE[25] a widely-used loss function in one-positive-multi-negative-pair contrastive representation learning, to pull close the representation of corresponding points, while push away representation of them and other point representations. The loss function is shown in Equation 3

$$\mathcal{L}_{CD} = -log(\frac{exp(f_p \cdot f_{p'}/\tau)}{\sum_{i=1}^{m} exp(f_p \cdot f_{p'_i}/\tau)}) \tag{3}$$

where f_p is the skeleton point in realworld garment, $f_{p'}$ is the corresponding point in simulation point cloud and $f_{p'_i}$ is other point in simulation point cloud.

As shown in figure 6, after model finetune, the performance of the model on realworld garment improve significantly. This is mainly because model is more adapted to the distribution of real world point cloud.

F Real-World Benchmark

Benchmarking and performance evaluation in robotic manipulation encounter challenges owing to the diverse range of applications and tasks, prompting research groups to select representative tasks and objects that are frequently inadequately specified and inaccessible to others, thereby impeding the ability to compare experimental results and interpret performance quantitatively, particularly in real-world scenarios. To address this issue, the implementation of a real-world benchmark is crucial, as it can not only narrow sim2real Gap but also provide a platform for researchers to directly compare algorithm performance. Although previous work has introduced real-world benchmarks, such as YCB [5] and furniture benchmark [18], they primarily focus on rigid bodies, lacking benchmarks specifically designed for deformable objects. **In this study, we introduce the first real-world benchmark for deformable objects and garments, facilitating the widespread usage of a standardized set of objects and tasks to enable easy comparison of results among research groups worldwide.**

F.1 Object and Data Set: Object Selection

Principle
We aimed to select objects that are frequently used in daily life, and we also reviewed the literature to consider objects that are frequently used in simulations and experiments. Several additional practical factors must be considered when formulating the proposed set of goals and tasks.

- ***Variety***
  The objects included are small in number, but ensure a great richness. Judging from the category of items, we roughly include tops, pants, skirts, socks, gloves, dolls, etc. Considering size, generally speaking, clothes occupy a larger area, followed by dolls, and then small items such as socks and gloves. Considering deformability, large items of clothing are the softest, can be stacked into various shapes, and have the highest deformability, followed by small items, which can only undergo simple changes because they are relatively small, while dolls are elastic but lack deformability. Grasping and manipulation difficulty was also a criterion: for instance, toys are well approximated by simple geometric shapes and relatively easy for grasp synthesis and execution, while garments have higher shape complexity and are more challenging for grasp synthesis and

https://doi.org/10.52202/079017-0379

execution. In addition, since we are doing a benchmark about garments, we have to carefully consider their characteristics: they are diverse and highly deformable, but the same type of garment often only differs in texture or color, and is very similar in structure and key points. This allows us to use a few objects to represent a category of garments, thereby ensuring the variety of our benchmark.

- ***Use***
  We included objects that are not only interesting for grasping but that also have a wide range of manipulation uses. Soft and highly deformable clothing is also suitable for many complex operations: such as hanging, folding, etc. The introduction of fluid allows us to simulate the interaction between some objects and fluids, such as washing and air-drying. In addition, we also included people, which allowed us to simulate the interaction between some objects and people, such as putting a scarf on someone. As mentioned above, these tasks are intended to span a wide range of difficulty, from relatively easy to very difficult.
- ***Durability***
  We aimed for objects that can be useful in the long term, and, therefore, avoid objects that are fragile or perishable. In addition, to increase the longevity of the object set, we chose objects that are likely to remain in circulation and change relatively little in the near future.
- ***Cost***
  We aimed to keep the cost of the object set as low as possible to broaden accessibility. We, therefore, selected standard consumer products, rather than, for instance, custom-fabricated objects, and tests. We buy all our clothes from Uniqlo and all our dolls from Disney.

After these considerations, the final objects were selected. You can see our object set in Table 8 and the corresponding suggested tasks in Table 9.

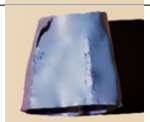
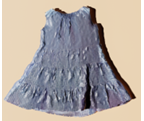
Table 9: **Suggested Manipulation Tasks for Different Garments.**

Object Category	Suggested Tasks
Tops/Pants/Shorts/Vests/Skirts	a. Washing and Drying b. Folding c. Stacking and Grabbing of clothes
Dresses/Coats	a. Washing and Drying b. Folding c. Stacking and Grabbing of clothes d. Hanging
Hats/Scarfs	a. Washing and Drying b. Folding c. Stacking and Grabbing of clothes d. Hanging e. Interacting with People: wearing corresponding objects on people
Deformable Objects	a. Grabbing and Placing
Small Items (Gloves/Stocks)	a. Grabbing and Placing b. Washing and Drying

F.2 Object Scans

Our scanning process is roughly divided into four stages: model wearing clothes, scanner scanning to obtain raw data, post-processing, and manual annotation of key points. The whole process can be referred to Figure 9.

Table 8: **Real-World Objects and Properties**. PBD stands for Position Base Dynamics, while FEM stands for Finite Element Method.

Number	Class	Object Name	Picture	Simulation Method	Branch	Number	Class	Object Name	Picture	Simulation Method	Branch
1	FEM Objects	Straw -berry Bear		FEM	Disney	11	Shorts	Blue Denim Shorts		PBD	Uniqlo
2	FEM Objects	Stitch		FEM	Disney	12	Shorts	White Shorts		PBD	Uniqlo
3	FEM Objects	Winnie Bear		FEM	Disney	13	Skirts	Blue Denim Shorts		PBD	Uniqlo
4	Coats	White Jacket		PBD	Uniqlo	14	Vests	White Cotton Vest		PBD	Uniqlo
5	Coats	Green Jacket		PBD	Uniqlo	15	Shorts	White Camisole Vest		PBD	Uniqlo
6	Coats	White Plush Jacket		PBD	Uniqlo	16	Dresses	Blue Dress		PBD	Uniqlo
7	Tops	White-long-sleeved T-shirt		PBD	Uniqlo	17	Shorts	White Blue Dress		PBD	Uniqlo
8	Tops	White-short-sleeved T-shirt		PBD	Uniqlo	18	FEM Objects	Blue Hat		FEM	Uniqlo
9	Pants	White Pants		PBD	Uniqlo	19	FEM Objects	Blue Socks		FEM	Uniqlo
10	Pants	Black Pants		PBD	Uniqlo	20	FEM Objects	Yellow Gloves		FEM	Uniqlo

https://doi.org/10.52202/079017-0379

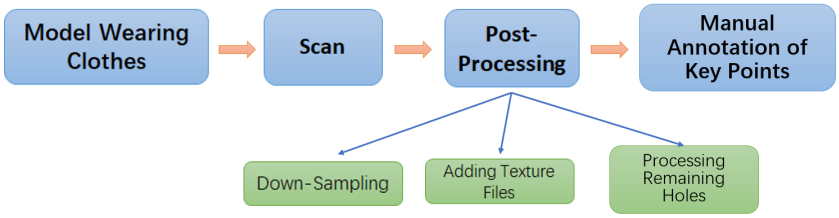


Figure 9: **Objects Scanning Process.**

- ***Model wearing clothes***
  There are two main ways to scan clothes: scanning them flat and scanning them while the model is wearing them. After careful consideration and constant experimentation, we chose the latter. This is because we use a large number of points to simulate clothing, so the wrinkles formed when the clothes are laid flat will cause the points to be unevenly distributed, thus forming many "holes". While when worn on the model, the appearance of wrinkles will be reduced, thus try to avoid this situation as much as possible.

- ***Scanning***
  We use a scanner to scan the object from multiple angles, and obtain a copy of the original point cloud data and grid data through the combination of the depth camera and the RGB camera. The general process can be referred to Figure 5.

- ***Post-Processing***
  During the post-processing process, we mainly did three things: down-sampling, adding texture files, and processing remaining holes. The point set obtained by the initial scan has too many points and is difficult to support with ordinary computing power, so we performed down-sampling to generate a file that can retain the main features and have a moderate number of points. We use Meshlab for down-sampling. The original scale is about 100 million points and 300 million faces. After down-sampling, it can reach about 10,000 points and 30,000 faces. We use MTL files (Material Library File) to add material attribute information. MTL is a material library file used to describe the material information of objects. It is usually used in conjunction with an OBJ file to apply material properties such as texture and color to the OBJ model. In this step, we mainly implemented some visual textures, such as patterns, colors, etc. The physical texture is achieved through different simulation methods. In addition, there are still some “holes” in the processed point set, which we repaired manually.

- ***Manual annotation of key points***
  Since we use a large number of points to simulate objects, it is necessary to mark some key points and edges to indicate important features. For example, for tops, we will mark the sleeves, neckline, hem, etc. These locations are often the key parts for clothing operations. For dolls, we will mark arms, legs and other parts that are easy to grasp. Figure 10 gives some examples for reference.

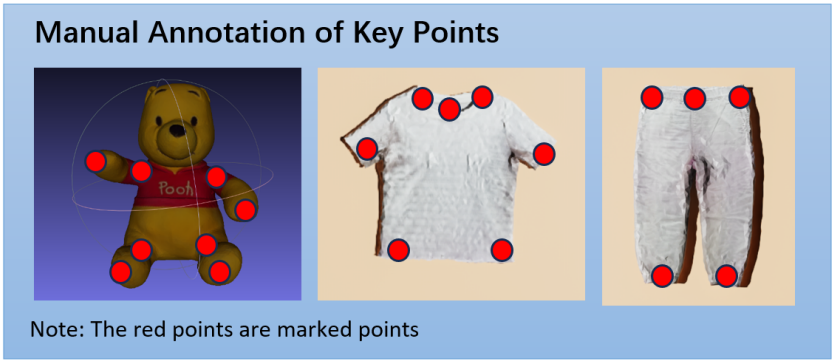


Figure 10: Manual annotation of key points: examples

https://doi.org/10.52202/079017-0379

F.3 Protocol Benchmark Guidelines

We use the protocol and benchmark templates mentioned in [72]. To both provide more concrete samples of the types of task definitions that can be put forward as well as specific and useful benchmarks for actually quantifying performance, we have developed some example protocols: clothes-hanging protocol, scarf-wearing protocol.

- ***Clothes-hanging protocol and benchmark***
  When dealing with flexible clothing, hanging clothing is always a popular task. The protocol uses the hanger, clothes suitable for hanging of our model set. The clothes are initially laid flat on the platform, and the robot is expected to grab the key points of the clothes, pick them up, and finally hang them on the hangers. The benchmark scores the performance of the robot by evaluating whether the hanging point is reasonable and the stability of the hanging clothes (whether they are easy to fall off). We applied this benchmark to Franka.
- ***Scarf-wearing protocol***
  Dressing people with robots has always been a difficult subject. In this example, we chose the easier and more manageable task: wearing a scarf. This protocol uses the scarf from the model set, and a person from the Issac Simulator. The scarf is initially laid flat on the platform, and what the robot has to do is to pick it up and wrap it around the person's neck, and finally adjust it to a suitable state. The benchmark scores the performance of the robot by evaluating whether the final state of the scarf is stable (we can test it by adding wind to see if the scarf will blow off quickly), whether the remaining length of the scarf on both sides is similar, and whether the scarf fits the person's neck (rather than loosely packed). We applied this benchmark to Franka.

G Task

G.1 Task category

Garment-Garment. This category focuses on fundamental garment manipulation, including tasks like folding and unfolding single garments, as well as interactions between multiple garments such as retrieving items from clothes piles. Tasks in this category include folding, unfolding (pick and place), and unfolding (fling).

Garment-Fluid. Tasks in this group concentrate on the interaction between garments and fluids as well as flow, where trajectory dynamics play a crucial role. This category of tasks includes washing clothes in a basin, rinsing clothes under running water, and drying clothes with a hairdryer. In this category, we specifically introduced the interaction between the robot manipulating objects and fluid flow, including both water flow and air flow.

Garment-FEMObjects. We mainly focus on the exploration of tasks involving deformable interactions, such as using a sponge to clean dirt off clothes or packing hats and tops together. Some simple tasks involving the manipulation of deformable objects are also included, such as using a dexterous hand or gripper to grab plush toys.

Garment-Rigid. Common interactions between clothing and rigid bodies, such as hanging clothes or putting them into a washing machine, require precise grasp point selection and trajectory planning.We also introduced articulated objects such as cabinet drawers and clips to perform garment-related tasks, such as taking clothes out of a wardrobe.

Garment-Avatar. Dressing tasks pose the greatest challenge, as they require understanding of human intention and safe collaboration with humans. Some representative tasks include putting a scarf on a person and placing a hat on their head. More advanced tasks involve dressing a person in a jacket or a T-shirt.

G.2 Long-horizon task

Organizing clothes. This task comprises several stages, including retrieving tops or trousers from a clothes pile, unfolding them using fling, folding the clothes, and placing them in the wardrobe.

Wash clothes. This task involves several stages: retrieving a hat from the cabinet, washing the hat in the basin, using a hairdryer to dry the hat, and placing the hat on a hanger.

 https://doi.org/10.52202/079017-0379

Make up table. The task of setting the table involves several steps: firstly, retrieving the tablecloth from the box, laying it flat, spreading it onto the table, smoothing it out, and finally adjusting its position. Note here we need the use of mobile robot like mobile franka.

Dress up. This task involves putting a scarf on someone, placing a hat on someone's head, and dressing someone in a T-shirt.

H MoveIt

We adopt MoveIt, an open-source state-of-the-art robotic manipulation framework, to provide support for real-world trajectories planning and obstacle avoidance. To record the trajectories generated by MoveIt and adapt visual models in the simulator to the trajectories could bridge the sim2real gap to some extent. In this section, we introduce a smooth, lightweight and responsive signal pipeline implementation to transfer real-world joint parameters to the simulator.

As a robotic manipulation platform, MoveIt is built on top of ROS(Robot Operating System) and integrates with various ROS components. MoveIt provides a series of comprehensive manipulation interfaces, including collision-free motion planning, kinematics computation, collision detection, etc. Moreover, real-world data collected from sensors like cameras and lidars can be fed into MoveIt, allowing for dynamic obstacle avoidance. Once the trajectories are generated by MoveIt, we publish the computed joint states through ROS, which transfers the trajectory to the Franka controller, FrankaPy. FrankaPy is a modular control stack that provides a customizable and accessible interface to the Franka robot. Utilizing MoveIt and FrankaPy, this pipeline enables Franka to devise a collision-free path and guide the gripper to the target posisition using vision detectors, while publishing the joint parameters to the ROS server. The simulator then subscribes the joint states and moves the Franka model accordingly.

I Teleoperation

Teleoperation serves as a direct method to acquire human demonstrations for model training. To accurately and smoothly track human hand motions has been proved advantageous in related works recently. However, the proliferated fine-grained tracking requirements, along with sparse and diverse dexterous hand models and environment settings, have posed a challenge towards teleoperation systems. Compared to controller-based models, we utilize the vision-based motion detection module, Leap Motion, to efficiently record human hand poses and then retarget hand poses to the dexterous robot hand. More Formally, our teleopertaion systems can be described as below:

(i) **Leap Motion hand pose detection module**, which predicts the wrist position, the hand spatial direction and finger poses from the infrared camera stream.

(ii) **retargeting module**, which converts the wrist position and finger poses recorded by Leap Motion to the arm end effector position and dexterous hand parameters.

(iii) **motion generating module**, which produces accurate, responsive and high-frequency signals for the robot model that in the simulator and in the real world simultaneously.

In our work, we implement teleoperation for Universal Robot mounted with Shadow Hand and Franka. One can control the posture of the robot by adjusting attitude and position of the hand over the detector. In particular, the open state of the gripper of Franka can be controlled by the opening and closing of the thumb and index finger.

I.1 Leap Motion Detection Module

The Leap Motion Controller is a small USB device that can be placed on the desktop. Utilizing two 640x240-pixel near-infrared cameras, it captures a roughly hemishperical area in the distance of approximately 60 cm, typically at 120Hz. The internal algorithms then translate the received raw spatial data to 27 distinct hand elements, which includes the palm normal vector, the hand direction, the wrist position and 24 finger joint positions. The detailed hand elements are shown in the figure 12:

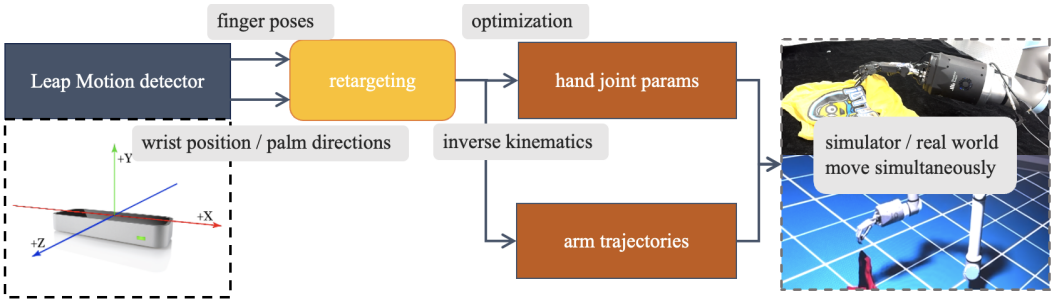


Figure 11: **Details of Our Teleoperation System.**

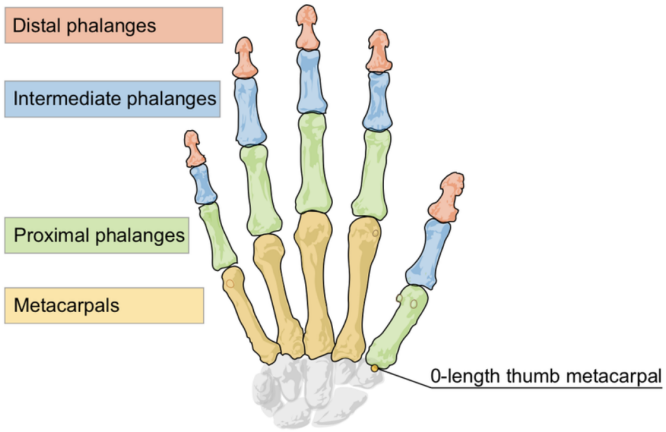


Figure 12: **The Hand Model Used in Leap Motion**, each knuckle joints' spatial position is computed by retargeting algorithm and broadcasted though ROS Message.

I.2 Hand Pose Retargeting

The procedure of hand pose retargeting is two-fold: first, we map knuckle positions to hand joint parameters; second, we compute a trajectory to smoothly move the robot arm to the recorded wrist position and direction. The finger knuckle positions captured by Leap Motion cannot be directly fed into robot models due to the discrepancy between the dexterous hand joint angular parameters and the knuckle positions. The mapping algorithm that converts knuckle positions to joint parameters is often formulated as an optimization problem, which can be described as

$$\min_{q_t} \sum_{i=0}^{N} \left\| \alpha v_t^i - f_i(q_t) \right\|^2 + \beta \left\| q_t - q_{t-1} \right\|$$

where q_t represents joint parameters of the dexterous robot hand at time step t. f_i is the i-th forward kinematic function which takes the joint angles as input and computes the knuckle positions. α is a scaling factor to alleviate the size discrepancy between different human operators and the robot hand model. Additionally, we observe adjacent N frames and add hyperparameter β to improve temporal smoothness and consistency.

To compute the arm trajectory, we adopt a slightly tweaked inverse kinematics approach, which is popular to determine the joint parameters in the trajectory, given the URDF file (Unified Robotics Description Format) of the robot and the desired configuration. URDF is a file format that describes the physical properties and 3D model of the robot, including joints, motors, articulation configurations, etc. In empirical experiments, we find that even slight hand movement or vibration can trigger a significant and prolonged changes in arm posture. To address this problem, we implement a rate limit on target changing in neighboring frames.

https://doi.org/10.52202/079017-0379

It is notable that this workflow is applicable to a vast range of grasp-based robots. Particularly, to control the open state of the gripper of Franka, we measure the distance of the thumb and index finger and establish a distance threshold in implementation.

I.3 Motion Generation

We integrate ROS for communication. ROS is a open-source framework that provides a series of library which are designed for multi-robots scenarios. In our implementation, after the generation of hand pose and wrist position, the computed joint parameters are transferred through the ROS to the simulator and the non-virtual robot simultaneously.

J Limitation

While this work adopts state-of-the-art simulation for different garments, there still exists the gap between the dynamics and kinematics of garments in simulation and the real world.

K Broader Impact

This work paves solid way to future home-assistant robots in diverse garment tasks for industry. We haven't observed negative potential impacts.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Section 1

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section J

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

https://doi.org/10.52202/079017-0379

Justification: no theoretical result

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Section B

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: As we claimed in abstract, we will release our code as soon as possible

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4, 5 and 7

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Section 7. We calculate of the variance of scores of the policy in different initial settings.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

https://doi.org/10.52202/079017-0379

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section B

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Section K

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Section K

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: no data or models that have a high risk for misuse

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: A

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

https://doi.org/10.52202/079017-0379

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: Section 3.2

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: no crowdsourcing experiments and research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: no experiments with potential risks for study participants

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.