

Supplementary Materials

TypeTele: Releasing Dexterity in Teleoperation by Dexterous Manipulation Types

1 TypeTele System Details

1.1 Dexterous Manipulation Type Library

Visualization of Type Library

We construct a dexterous manipulation type library using the taxonomy, based on prior grasp type work [1, 2, 3, 4, 5], and extending it based on the structure of dexterous hands [6] and manipulation tasks. The visualization of the library is illustrated in Figure 1.



Figure 1: Visualization of Dexterous Manipulation Type Library.

Annotation Information of Dexterous Manipulation Types

Each manipulation type is annotated with descriptive information to characterize its posture and functionality, which facilitates retrieval. The annotated attributes include: *hand posture*, *manipulable object categories*, *contact parts on the object*, *the geometry of these parts*, *grasp direction*, and *intended manipulation purpose*. Examples of the annotated information are shown in Figure 2.

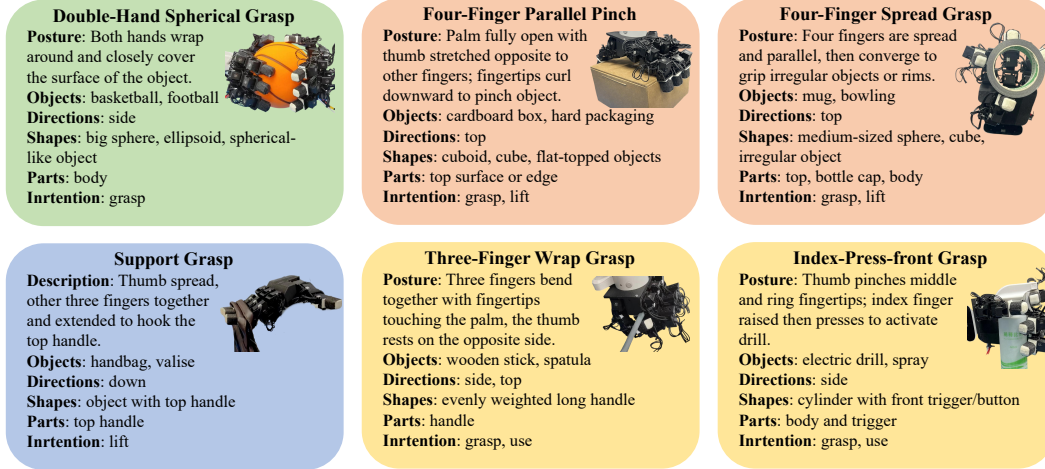


Figure 2: Visualization of the annotations examples of types in the library.

1.2 MLLM-assisted Type Retrieval Module

We employ GPT-4o [7] to retrieve the most suitable dexterous type for current task. The details of prompts are as following:

```

System Prompt:
You are a robotic manipulation expert. When given a user goal and an image
of tools or ingredients, your job is to:
(1) Decompose the task into clear manipulation steps.
(2) Assign a suitable grasping type for each hand (left/right) in every
step based on the provided grasp type library.
(3) Format your response in this structured way:
The task is divided into N steps:
Step 1: [describe the subtask]
Step 2: [describe the subtask]
...
The types in each step are:
Step 1: Left type: [grasp type name] Right type: [grasp type name]
Step 2: Left type: [grasp type name] Right type: [grasp type name]
...
User Command: I want to [describe the whole task].

```

Example of Output

User Command

```

I want to prepare a pancake with tomato sauce and a glass of water for
breakfast.

```

GPT-4o

```

The task is divided into 3 steps:
Step 1: Pick up the pan with the pancake using the right hand, and hold and
squeeze the tomato sauce bottle using the left hand to apply sauce.
Step 2: Keep holding the pan with the right hand, and use a spatula with
the left hand to transfer the pancake into the bowl.
Step 3: Hold the water pitcher with the left hand and the cup with the
right hand, then pour water into the cup.
The types in each step are:
Step 1:

```

52	Left type: Thick Cylinder Grasp (for squeezing the sauce bottle)
53	Right type: Three-Finger Load-Bearing Wrap Grasp (for lifting the pan by
54	the handle)
55	Step 2:
56	Left type: Three-Finger Wrap Grasp (for using the spatula)
57	Right type: Three-Finger Load-Bearing Wrap Grasp (continue holding the pan)
58	Step 3:
59	Left type: Curved Handle Grasp (for holding the pitcher)
60	Right type: Thick Cylinder Grasp (for holding the glass)

62 1.3 Adjustment for Dexterous Manipulation Types

63 To enhance the applicability and adaptability of our system, TypeTele supports the adjustment of
64 dexterous type. Users can perform type adjustment to apply offsets to the position or rotation of
65 each fingertip via the keyboard or voice commands during teleoperation.

66 Specifically, the system first obtains the initial fingertip position and rotation of the origin type
67 through forward kinematics. Then the system applies the offsets to the fingertip, and uses the new
68 fingertip position and rotation to derive the adjusted joint angles via inverse kinematics. To ensure
69 the adjusted type remains as close as possible to its initial pose, the system uses the joint angles of
70 origin type as the initial solution for the inverse kinematics solution, which can avoid unexpected
71 changes in the resulting posture.

72 1.4 Kinesthetic Teach Module

73 Although the types provided in our type library can handle the most of everyday applications, we
74 offer a teaching mode for the creation of new type for special cases and unique user needs. This
75 allows users without robotics expertise to intuitively and conveniently create dexterous types.

76 We implemented the teaching mode using admittance control and motor backdrivability. The for-
77 mula for admittance control is as follows:

$$M\ddot{x}(t) + B\dot{x}(t) + Kx(t) = F_{\text{ext}}(t) \quad (1)$$

78 In this equation, $x(t)$ denotes the position of the control output, $\dot{x}(t)$ and $\ddot{x}(t)$ represent the velocity
79 and acceleration respectively, and $F_{\text{ext}}(t)$ is the external force applied to the robot's end-effector.
80 The parameters M , B , and K correspond to the virtual mass, damping, and stiffness, respectively.

81 Here, we estimate the external force using the current magnitude and positional deviation of the
82 dexterous hand's motors. We also incorporate the motor's velocity information to give the motion a
83 certain degree of inertia, making the teaching process smoother.

84 1.5 Robot Control

85 To enhance the control fidelity and operational fluidity of the robotic arms during both teleoperation
86 and imitation learning inference, we implemented several key methodological improvements:

87 **(1) Multi-threaded Device Communication:** We have adopted a multi-threaded approach for de-
88 vice communication. Each distinct data stream – including RGB-D imagery, cartesian poses of the
89 two end-effectors, and joint angles of the two robotic hands – is managed by an independent thread.
90 This architecture ensures that when the main thread requires specific information, it can be provided
91 instantaneously, thereby circumventing delays typically associated with data acquisition operations.

92 **(2) Uniformly Accelerated Motion for Velocity Control:** For both translational and rotational
93 velocity control, we have applied uniformly accelerated motion profiles. This strategy guarantees
94 that velocity changes are smooth and devoid of abrupt transitions. Consequently, the robotic arm's
95 movements are exceptionally fluid, and this approach also mitigates jitter stemming from natural
96 human hand tremors or sensor inaccuracies.

97 **(3) Dynamic Speed Control for Rotation:** A dynamic speed control scheme has been implemented
 98 for rotational movements. When the current orientation is significantly distant from the target orien-
 99 tation, the rotational speed will be increased, enabling the robotic arm to rapidly converge towards
 100 the desired direction. Conversely, as the current orientation approaches the target, the rotational
 101 speed will be reduced. This allows the operator to perform precise, fine-grained rotational adjust-
 102 ments.

103 **(4) Dedicated Asynchronous Robot Control Thread:** The robotic arms are controlled by a dedi-
 104 cated, separate thread, ensuring asynchronous operation. The main thread focuses solely on trans-
 105 mitting the target pose to this robot control thread. Subsequently, the robot control thread governs
 106 the robotic arms at a consistent control frequency. This approach guarantees stable and smooth
 107 robotic arm control, even when the main thread’s frame rate fluctuates or varies, such as during
 108 transitions between teleoperation and imitation learning inference with their differing frame rates.

109 1.6 Details of Imitation Learning

110 We adopt a diffusion-based imitation policy to learn from expert demonstration data, following [8].
 111 The observation input consists of single-view point clouds $x_i \in \mathbb{R}^{N \times 3}$ and robot proprioceptive
 112 inputs $x_p \in \mathbb{R}^p$. Specifically, we downsample $N = 4096$ points from the raw depth maps. The
 113 proprioceptive input ($p = 44$) includes the Cartesian poses of both robot arms and the joint angles
 114 of the two dexterous hands.

115 The point clouds are encoded using a pyramid convolutional encoder [8], while the proprioceptive
 116 inputs are processed via a multilayer perceptron (MLP). We define the observation horizon as t_o and
 117 the action horizon as t_a . In our setup, we adopt a fixed total horizon length of $t_o + t_a - 1 = 15$. For
 118 tasks requiring longer-term reasoning or delayed consequences, we use longer observation horizons
 119 (e.g., $t_o = 6$ or 8) and shorter action horizons (e.g., $t_a = 10$ or 8), while for reactive or short-horizon
 120 tasks, we opt for shorter observations (e.g., $t_o = 3$ or 4) and correspondingly longer action horizons.
 121 This enables a flexible temporal encoding of task-relevant information, tailored to the nature of each
 122 behavior. All features are used as conditional inputs to predict the noise associated with the robot
 123 action $a \in \mathbb{R}^{k_a}$, where k_a denotes the action dimension specific to the task. The training objective
 124 minimizes the denoising score matching loss, formulated as:

$$\mathcal{L} = \mathbb{E}_{\mathbf{a}_0, \epsilon \sim \mathcal{N}(0, \mathbf{I})} \left[\|\epsilon - \epsilon_\theta(\mathbf{a}_t, t)\|^2 \right], \quad (2)$$

125 where $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ is the Gaussian noise. The network ϵ_θ is trained to predict the added noise given
 126 the noisy action \mathbf{a}_t and the timestep t . We employ DDIM [9] for inference sampling.

$$\mathbf{a}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \left(\frac{\mathbf{a}_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(\mathbf{a}_t, t)}{\sqrt{\bar{\alpha}_t}} \right) + \sqrt{1 - \bar{\alpha}_{t-1}} \cdot \epsilon_\theta(\mathbf{a}_t, t) \quad (3)$$

127 where $\bar{\alpha}_{t-1}$ and $\bar{\alpha}_t$ are the cumulative noise schedule coefficients at time steps $t - 1$ and t , respec-
 128 tively.

129 2 Experiments Details

130 2.1 Details of Tasks

131 **Task 1: Pick and Place.** Task 1 is a fundamental task that requires picking up a tennis ball on the
 132 table and placing it into a basket.

133 **Task 2: Collect and Store.** Task 2 focuses on the integrated capabilities of the system. Task 2
 134 requires collecting three objects from the table and placing them into the basket in the following
 135 order: doll, broom, and basketball. For TypeTele, the operator can use voice commands to switch
 136 types for objects with different geometric shape during teleoperation.

137 **Task 3: Handover.** Task 3 evaluates the system’s bimanual coordination capabilities and grasp
 138 robustness. In this task, the left hand is required to pick up a can from a stand and then hand it over
 139 to the right hand.

Task 4: Pouring from Pan. Task 4 requires stably grasping a pan and then pouring the contents of the pan into the basket. The difficulty of this task lies in the need for the hand to firmly grip the pan’s handle to prevent tilting or dropping during the pouring process.

Task 5: Use Scissors. In Task 5, the right hand is required to hold a strip of paper while the left hand uses a pair of scissors to cut through it. The task is considered successful if the lower part of the paper strip is completely severed in a single cut.

Task 6: Spray Water. Task 6 requires grasping a spray bottle and then pressing the trigger to spray water. The task is considered successful if a stream of water is sprayed out.

Task 7: Use a Heavy Kettle. Task 7 evaluates the ability to operate under extreme weight. Task 7 requires firmly gripping the handle of a watering kettle filled with water, lifting the watering can, and then pouring water into a bowl.

Task 8: Opening a Large Box. Task 8 evaluates the ability to manipulate objects of significant size. In this task, the left hand is used to open a large box, followed by the right hand retrieving the object contained within.

Task 9: Grasp Two Objects. Task 9 aims to fully leverage the dexterity of the dexterous hand. This task requires using one hand to grasp two objects, first a water cup and then a cylinder.

2.2 Additional Experiments

We conducted additional experiments to further evaluate our teleoperation system through a user study involving five participants with varying levels of prior teleoperation experience. Each participant was instructed to complete an identical task (grasping the handle of a frying pan) using both TypeTele system and a retargeting-based baseline. In order to mitigate the influence of learning effects, three participants used the TypeTele first, while the remaining two began with the baseline, and none were informed which system was the TypeTele and which was the baseline. Each system was tested in five trials per participant, during which we recorded success rate and Average Time per Success. The Average Time per Success is calculated by dividing the total time spent across all trials by the number of successful trials. This metric reflects the average amount of time required to obtain a single successful execution, capturing both task efficiency and failure overhead.

After completing all the tests, each participant completed a questionnaire that evaluated both systems in four dimensions: accuracy, responsiveness, ease of use, and user confidence. Each dimension was rated on a scale of 0-10. We then computed the average score for each dimension across all participants for both systems. The results are as follows:

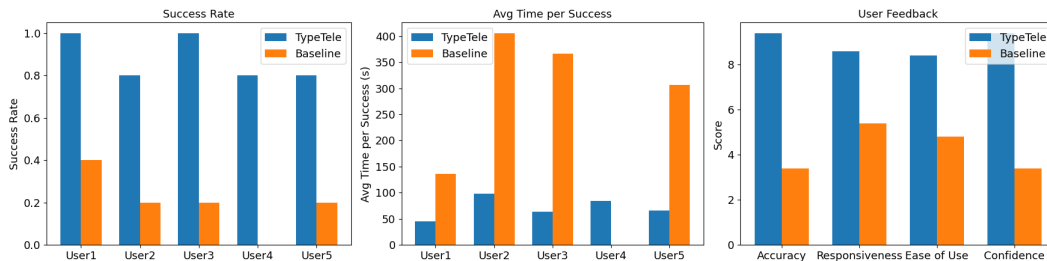


Figure 3: Results of User Study.

Experimental results demonstrate that the TypeTele system significantly outperforms the retargeting-based baseline across both objective and subjective measures. On average, TypeTele achieved a task success rate of 88%, compared to only 20% for the baseline. Participants also completed tasks faster using TypeTele. Subjective ratings further support these findings: TypeTele received higher scores across all four dimensions—accuracy (9.4 vs 3.4), responsiveness (8.6 vs 5.4), ease of use (8.4 vs 4.8), and user confidence (9.4 vs 3.4). These results indicate that TypeTele not only improves task performance but also delivers a more satisfying and trustworthy user experience.

References

- [1] T. Feix, J. Romero, H.-B. Schmiedmayer, A. M. Dollar, and D. Kragic. The grasp taxonomy of human grasp types. *IEEE Transactions on human-machine systems*, 46(1):66–77, 2015.
- [2] M. R. Cutkosky et al. On grasp choice, grasp models, and the design of hands for manufacturing tasks. *IEEE Transactions on robotics and automation*, 5(3):269–279, 1989.
- [3] F. Krebs and T. Asfour. A bimanual manipulation taxonomy. *IEEE Robotics and Automation Letters*, 7(4):11031–11038, 2022.
- [4] H.-S. Fang, H. Yan, Z. Tang, H. Fang, C. Wang, and C. Lu. Anydexgrasp: General dexterous grasping for different hands with human-level learning efficiency. *arXiv preprint arXiv:2502.16420*, 2025.
- [5] J. Chen, Y. Chen, J. Zhang, and H. Wang. Task-oriented dexterous hand pose synthesis using differentiable grasp wrench boundary estimator. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5281–5288. IEEE, 2024.
- [6] K. Shaw, A. Agarwal, and D. Pathak. Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning. *arXiv preprint arXiv:2309.06440*, 2023.
- [7] A. Hurst, A. Lerer, A. P. Goucher, A. Perelman, A. Ramesh, A. Clark, A. Ostrow, A. Welihinda, A. Hayes, A. Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.
- [8] Y. Ze, Z. Chen, W. Wang, T. Chen, X. He, Y. Yuan, X. B. Peng, and J. Wu. Generalizable humanoid manipulation with improved 3d diffusion policies. *arXiv preprint arXiv:2410.10803*, 2024.
- [9] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.