DexH2R: A Benchmark for Dynamic Dexterous Grasping in Human-to-Robot Handover

Supplementary Material

A. Dataset

A.1. Object Selection in Our Dataset

Considering the safety during human-to-robot handover, the difficulty of annotating object poses, the commonality of objects, and the objects' geometric irregularity, we selected 40 objects from the RealDex [26] dataset for our study. To further enhance the diversity of objects in our dataset, we acquired an additional 16 new objects and utilized the EinScanPro+ [2] high-precision 3D scanner to obtain high-quality meshes for these objects. Consequently, our dataset comprises a total of 56 objects, which are carefully chosen to cover a wide range of common items encountered in daily life.

A.2. Details for Camera Capture Settings

Azure Kinect Capture Settings Four Kinect cameras are placed at the corners of the workspace to record RGBD information of the interactive scene, which aids in scene understanding and subsequent processing of object point clouds and pose annotation. These cameras are set to record at 15Hz with a resolution of 1080P, using the WFOV unbinned depth recording mode.

Intel RealSense Capture Settings Two RealSense cameras are mounted at the end of the robotic arm using custom-designed 3D-printed fixtures. These provide ego-view RGBD data, with one offering a wrist view and the other positioned slightly above the robotic hand to provide a first-person top-down perspective. Both are set to record at 15Hz with a resolution of 640×480.

ZCAM E2 Capture Settings Twelve ZCAM cameras are arranged in a quarter-sphere layout on the human-giver side of the workspace. These cameras capture high-quality RGB data, recording at 30Hz with a 4K resolution. The diverse viewpoints from these cameras are valuable for subsequent annotation of hand poses.

Recording Duration Setting To minimize manual annotation effort, we aimed to capture as many handover interactions as possible within a single recording session. However, hardware limitations such as memory and bandwidth restricted continuous recording durations. After testing, we set each session to 4 minutes to balance data density and system stability.

A.3. Calibration and Synchronization

Accurate calibration and precise synchronization are critical for ensuring the integrity and usability of multimodal datasets, particularly in tasks involving 3D perception, motion tracking, and human-robot interaction. In our setup, calibration establishes spatial consistency across heterogeneous sensors, while synchronization guarantees temporal alignment of data streams. The coordinate system in the ROS of the dexterous manipulation system is adopted as the global frame to ensure seamless alignment between multimodal sensor streams, eliminate cross-device transformation drift, and maintain consistency with robotic control pipelines.

The Global Coordination System To facilitate the extrinsic calibration of devices within the system, the coordinate system recorded in the ROS bag file from the robotic arm is designated as the global coordinate system. The origin is set directly below the robotic arm's chassis, with one edge of the desktop aligned parallel to the Y-axis and the other edge parallel to the X-axis. The interaction area lies in the positive Z-axis direction, ensuring that all point cloud data are confined to the quadrant where both the X-axis and Z-axis are positive. This setup provides a consistent reference frame for calibration across the system.

Camera-camera Calibration The intrinsic parameters of the cameras are sourced from the manufacturer, while extrinsic parameters are manually calibrated based on their viewing perspectives. The 4 Azure Kinect cameras are calibrated using markers on a desktop calibration rig and the robotic arm, with corresponding point pairs identified across adjacent views. For the 12 ZCAM E2 cameras, extrinsic parameters are estimated using a standardized checkerboard-based procedure. For the 2 ego-view RealSense cameras, real-time extrinsic parameters are dynamically updated by fusing the robotic end-effector's pose from forward kinematics and the pre-calibrated transformation between the camera and forearm mount. Initial calibration involves annotating fiducial markers on a shared rig across synchronized RealSense and Kinect point clouds, enabling precise camera-to-global transformation.

Details for Fixed View Camera Calibration For our system, all cameras with a fixed view are aligned to the coordinate frame of Kinect 0 first, achieving robustness across the sensor network. The coordinate system alignment between ZCAM and Kinect is achieved by computing an ICP-based transformation matrix using the ZCAM-annotated MANO mesh and the Kinect-derived segmented hand point cloud.

Details for Ego-view Camera Calibration The calibration of the 2 Intel RealSense RGB-D cameras leverages the fixed relative pose between the RealSense and the robotic

forearm. First, a transformation matrix (RealSense to forearm) is computed by aligning the RealSense and Kinect point clouds using the calibration box. Specifically, the robotic arm is repositioned so that both the RealSense and Kinect capture the calibration box. By manually identifying point pairs, we compute the transformation matrix from RealSense to Kinect, and subsequently to the global coordinate system. Simultaneously, a bag file is recorded to calculate the global transform of the robotic forearm (forearm to global) using ROS forward kinematics. The transformation matrix from RealSense to forearm is then derived by multiplying the inverse of the forearm-to-global matrix with the RealSense-to-global matrix. This cascaded transformation ensures accurate and temporally consistent calibration of the ego-centric RealSense cameras.

Robot-camera Calibration To establish precise spatial consistency between the dexterous manipulation system and all vision system frames, we formulate robot-camera calibration as a point cloud registration problem. By recording a ROS bag of the stationary poses of the robotic arm and ShadowHand, we acquire their ground-truth mesh. Concurrently, 4 calibrated Kinect cameras capture scene data, from which we segment point clouds corresponding to the robotic components. The transformation matrix between Kinect and global frames can be computed by registering the extracted point cloud to the ground-truth mesh via the Iterative Closest Point (ICP) algorithm [5]. By systematically aligning all fixed-view cameras to the Kinect reference frame and deriving ego-centric transforms from kinematic chains, the entire sensor network achieves spatial coherence. Unified global coordinates enable robust cross-sensor data fusion, critical for trajectory reconstruction, grasp stability assessment, and handover policy learning.

Synchronization To address the challenges of multi-device synchronization and data integrity in our heterogeneous camera setup, we employ a multi-computer parallel recording architecture to capture sensor streams. This design ensures both accuracy during data acquisition and low latency during disk writing. Prior to formal data collection, we initiate a synchronization protocol that all cameras record a high-frequency multi-color LED panel with temporally encoded patterns. The ROS timestamps and camera frames are then manually annotated using visualization tools to identify synchronization anchors. This approach ensures precise timestamp alignment and synchronization across all modalities, closely reflecting real-world conditions.

A.4. Details for Data Processing

Time Segmentation During post-processing, we annotated three critical frames for each handover event: the initiation frame marking the moment when the human hand first contacts and attempts to lift the object, the stabilization frame indicating when the robotic hand securely grasps the object with minimal motion, and the termination frame captured just before the object fully leaves the robotic hand. This method of time segmentation not only allows for the extraction of valid motion data from a single recording but also enables the grasp frame and end frame to define an area where the pose data of the ShadowHand can be used to train a network for generating grasping pose of the ShadowHand.

Denoise Object Point Cloud Due to the hardware limitations of depth cameras, real point clouds invariably contain noise. To provide higher quality real data for downstream tasks, we employ DBSCAN [17] to denoise the cropped object point clouds, with specific parameters set at `eps=0.04` and `min_points=1000`, which have demonstrated commendable performance on our dataset.

A.5. Dataset Partitioning

The dataset is partitioned as follows: The training set consists of 2,888 trajectories involving 46 objects and 32 subjects. The test set contains 803 trajectories divided into three evaluation scenarios: (1) 136 trajectories featuring 10 unseen objects and 7 unseen subjects (complete novelty), (2) 369 trajectories with 46 seen objects and 4 unseen subjects (subject novelty), and (3) 298 trajectories with 5 unseen objects and 32 seen subjects (object novelty). For ablation studies, we sampled 100 trajectories from a 591-trajectory validation set, comprising two conditions: 46 seen objects with 3 unseen subjects, and 5 unseen objects with 32 seen subjects.

B. Evaluation Metrics

Penetration Depth The penetration depth measures how much the hand penetrates the object surface. For each hand point p_h, we find its closest object point p_o and compute the signed depth using the object normal n_o: **Success Rate 1** In Isaac Gym, we apply a single random force to the object grasped by the dexterous hand. If the object remains securely held without falling, we consider this a successful trial and record a Success Rate of 100%.

Success Rate 6 In Isaac Gym, we sequentially apply six distinct random forces to the object grasped by the dexterous hand. The trial is considered successful (Success Rate = 100%) if the object maintains a stable grasp throughout all force applications without falling.

$$d_{\text{pen}}(p_h) = \max(0, (p_o - p_h) \cdot n_o), \tag{1}$$

where the dot product determines if the hand point is inside the object. The overall penetration depth penalty is then computed as the mean of the maximum penetration depths across all hand points:

$$\text{pen\_depth} = \frac{1}{B} \sum_{i=1}^{B} \max_{p_h \in P_h} d_{\text{pen}}(p_h) \tag{2}$$

Diversity Diversity is calculated as the mean standard deviation of the grasping-pose parameters across successful grasps in millimeters.

C. Training Auto-regressive method

As shown in Eq. (3), the learning process is supervised using a loss function that measures the difference between the predicted and ground truth poses $\hat{q}_t$, along with a point cloud consistency loss that enforces alignment between the predicted dexterous hand's point cloud and the ground truth point cloud $\hat{P_t^h}$.

$$\begin{aligned} \mathcal{L}_{\text{MNet}} &= \mathcal{L}_{qpos} + \mathcal{L}_{pcd}, \\ \mathcal{L}_{\text{qpos}} &= \frac{1}{m} \sum_{t=t_n}^{t_{n+m}} ||q_t - \hat{q}_t||^2, \\ \mathcal{L}_{\text{pcd}} &= \frac{1}{m} \sum_{t=t_n}^{t_{n+m}} ||P_t^h - \hat{P_t^h}||^2. \end{aligned} \tag{3}$$

C.1. Approaching

Success Rate We define a trajectory as successful if the dexterous hand achieves goal pose alignment before the corresponding ground truth trajectory ends. For example, in a 100-step ground truth trajectory, if the approaching algorithm brings the hand into alignment before step 100, the trajectory is considered successful. This is because: (1) the system will automatically interpolate the hand to the pre-filtered goal grasping pose after alignment, and (2) the filtered grasping pose guarantees successful grasping. Therefore, reaching the alignment phase alone sufficiently demonstrates a successful approach.

Trajectory Length The trajectory length is the total 3D translational trajectory length accumulated by the dexterous hand during both the approaching phase and goal pose alignment phase.

Trajectory Frames The trajectory frames are defined as the sum of predicted frames during the approaching phase and interpolated frames during the goal pose alignment phase.

Penetration Depth The penetration depth is defined consistently with the criteria outlined in the Grasping Pose section. To compute this metric, we first calculate the average depth across the entire trajectory, which includes both the approaching phase and the goal pose alignment phase. We then determine the average penetration depth for the ground truth trajectories. The final penetration depth is obtained by subtracting the ground truth penetration depth from the calculated penetration depth of the trajectory.

Penetration Frames For each frame where the penetration depth is greater than zero, we increment the penetration frame count. This count aggregates data from both the approaching phase and the goal-alignment phase. Additionally, we compute the average penetration frames for the ground truth trajectories. The final penetration frame metric is derived by subtracting the ground truth penetration frames from the calculated penetration frames.

Safety Rate If the penetration frame is greater than zero, then the safety rate is 0%.

D. Ablation Study

D.1. Ablation Study on T_a and T_o

Model	T_o	T_a	Succ ↑	Safe ↑	Pen_dep↓	Pen_fr ↓	Infer_fr ↓	Traj_len
MotionNet	2	2	1.0	45.0	0.88	6.0	91.3	2.67
	5	2	16.0	48.0	1.02	6.14	89.7	1.29
	2	4	19.0	24.0	1.06	9.6	89.6	3.51
	5	4	20.0	26.0	1.07	8.69	89.4	2.01
	2	8	17.0	26.0	1.15	10.7	91.5	1.54
	5	8	17.0	23.0	1.08	12.5	90.4	1.60
	5	5	29.0	22.0	1.17	13.3	88.4	0.97
Diffusion Policy	2	2	3.0	71.7	1.65	4.5	91.2	1.43
	5	2	10.1	40.4	1.09	11.3	97.3	0.90
	2	4	2.0	69.7	1.49	5.0	91.2	1.15
	5	4	8.1	43.4	1.44	0.8	94.7	1.21
	2	8	3.0	41.4	1.43	9.3	91.3	1.36
	5	8	11.1	50.5	1.32	8.0	91.1	1.27
	5	5	14.1	45.5	1.45	8.4	93.2	1.28
Diffusion Policy 3D	2	2	26.3	38.4	1.36	8.4	93.9	1.59
	5	2	31.3	36.4	1.41	9.0	92.6	1.48
	2	4	19.2	46.5	1.35	8.2	94.3	1.49
	5	4	23.2	38.4	1.42	8.5	92.7	1.41
	2	8	12.1	46.5	1.37	8.7	90.8	1.38
	5	8	15.2	41.4	1.33	8.9	93.4	1.37
	5	5	22.2	41.4	1.33	9.0	93.0	1.40

Table 5. Ablation Study on Validation Subset (*itp* = 5 cm).

We sample 100 trajectories from the validation set to conduct an ablation study. In Tab. 5, we observe that for all three models, when T_a remains the same, a larger T_o leads to a higher success rate. This suggests that more historical information helps the model better predict the tracking trajectory by capturing motion patterns, reducing uncertainty, and distinguishing meaningful trends from noise, ultimately improving grasp accuracy. However, as historical information dominates, the **safe rate** may decrease. This effect is most pronounced in **Diffusion Policy**, while **MotionNet** and **Diffusion Policy 3D** maintain similar safe rates, demonstrating their robustness. Overreliance on history can make the model less adaptive to sudden changes, introduce response delays, and increase sensitivity to input noise. In contrast, the robustness of MotionNet and Diffusion Policy 3D suggests stronger temporal modeling or better generalization, helping them maintain stability. These findings highlight the trade-off between leveraging history for accuracy and balancing real-time observations for safety.

D.2. Ablation Study on Point Cloud Input

Since DP3 use point clouds as part of their observation input, we conducted an ablation study comparing raw object point clouds captured by a Kinect camera with clean point clouds sampled from the object mesh in Tab. 6 and

Method	objpcd_intact	Easy Mode (D=10cm)				
		succ ↑	safe ↑	pen_dep(cm) ↓	pen_fr ↓	infer_fr ↓
DP3	✓	68.3	48.6	0.73	4.7	97.9
	✗	66.3	50.1	0.73	5.0	96.8

Table 6. Comparison for Dynamic Motion Synthesis on test dataset (Easy Mode).

Method	objpcd_intact	Hard Mode (D=5cm)				
		succ ↑	safe ↑	pen_dep(cm) ↓	pen_fr ↓	infer_fr ↓
DP3	✓	28.7	33.8	0.80	8.8	105.9
	✗	27.1	33.7	0.84	9.3	103.7

Table 7. Comparison for Dynamic Motion Synthesis on test dataset (Hard Mode).

Tab. 7. Overall, DP3 achieved higher success rates when using clean point clouds as input.

E. Test on Real Robot

Our model, trained on real-world teleoperation data, can be directly deployed on real robots without any sim-to-real gap. For dynamic object grasping, we use four Kinect cameras to track the object's pose in real-time. The system inputs the object mesh, pose trajectory, and potential final grasp poses into MotionNet, which generates appropriate hand motion trajectories. These trajectories are converted to joint commands through real-time inverse kinematics. The hand transitions to the final grasp pose only when within 3cm of the intended position, ensuring precise grasping. The execution produces human-like, smooth movements that successfully grasp moving objects. After the human releases the object, the robotic hand maintains a secure grip, demonstrating the stability of our simulator-filtered grasp poses.

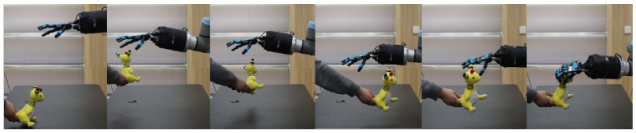

Figure 6. An example sequence of real-world dynamic grasping trajectory.