

Supplementary Materials – EquiBot: SIM(3)-Equivariant Diffusion Policy for Generalizable and Data Efficient Learning

Anonymous Author(s)

Affiliation

Address

email

1 A Detailed Discussion of Related Works

2 **Imitation learning from 3D inputs.** While many imitation learning works assume RGB images as
3 visual observations [1, 2, 3, 4], some works, similar to the design choice of our work, assume 3D
4 point clouds or depth inputs to their methods. Recently, Ze et al. [5] proposed *3D Diffusion Policy*, a
5 depth-only variant of diffusion policy for visuomotor policy learning. Their method is very similar
6 to our *DP* baseline, with two differences: (1) they use a simpler *DP3 encoder* in their work; (2) they
7 use a 2-layer MLP to encode the robot proprioceptive states before concatenating with the point
8 cloud representation.

9 In Figure 1, we show quantitative comparison
10 results between our method and baselines related to [5]. As in our main paper results, we
11 run 3 seeds of training runs, each for 2,000
12 epochs, for all experiments. We evaluate the
13 last 5 checkpoints for each training run and collect the final task reward for 10 episodes in
14 each evaluation. Comparisons show that [5]
15 displays similar performance as the *DP* baseline in the main paper, performing very well
16 in in-distribution setups and poorly in out-of-distribution setups.

21 We also show that [5] can be easily integrated
22 with our method by switching our PointNet-
23 based encoder to the *DP3 encoder*. In Figure 1,
24 we show a comparison between our method
25 and a variation of our method with a modification of the *DP3 encoder* to make it $SO(3)$ -
26 equivariant. Results show that the *DP3* variant
27 has slightly lower but comparable performance
28 as our method in the Cloth Folding task.

30 **Equivariant diffusion architectures.** Prior works have integrated equivariance in diffusion models
31 in various non-robotics domains, including molecule generation [6, 7, 8] and drug design [9]. Some
32 works have attempted to integrate equivariant architectures in diffusion models for robotics [10, 11].
33 Diffusion-EDFs [10] take point cloud observations as input and output a single target end-effector
34 pose. This formulation makes the work only applicable to pick-and-place tasks. EDGI [11] proposed
35 an open-loop policy and showed it in simple 2D tasks only. Compared to prior works, our work
36 proposes an equivariant diffusion policy that supports closed-loop 3D manipulation tasks.

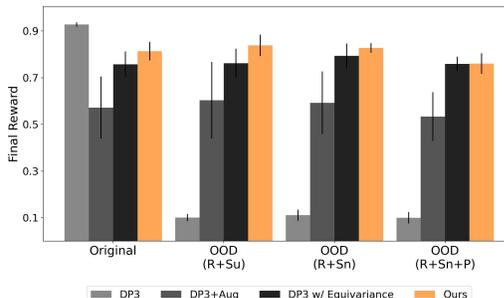


Figure 1: **Comparisons with DP3-related architectures in the Cloth Folding task.** We compare with three different baselines: (1) *DP3* is a variation of the *DP* baseline with the PointNet-based encoder replaced by the *DP3 encoder* proposed in [5], with the code of *DP3 encoder* copied verbatim from the public codebase; (2) *DP3+Aug* is a variant of the *DP3* baseline trained with augmentations that are the same as the *DP+Aug* baseline in the main paper; (3) *DP3 w/ Equivariance* is the integration of *DP3* into our method.

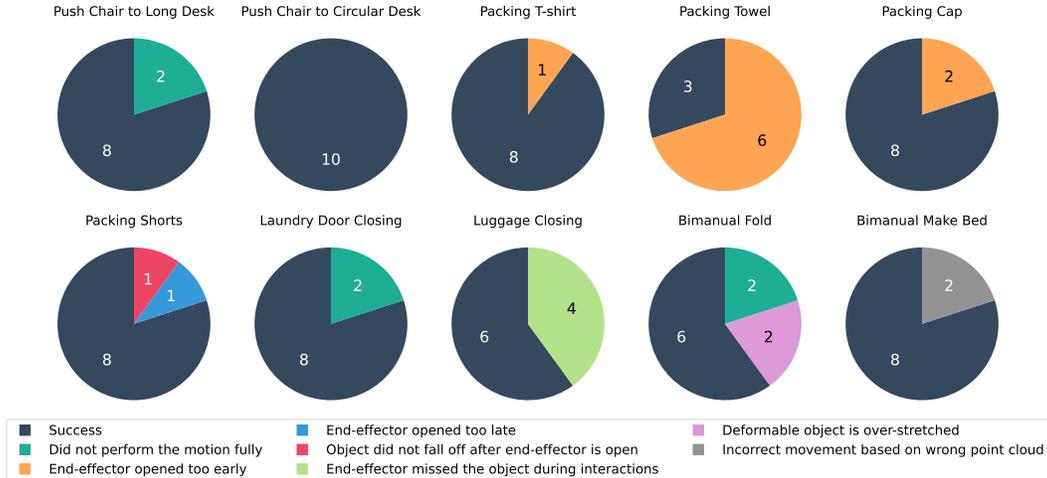


Figure 2: **Failure cases of our method during real robot executions.** The pie charts show failure breakdown in every real robot task variation. The navy color denotes success, while other colors denote different types of failures. Each pie chart shoes a total of 10 trials since we run 10 episodes per evaluation.

37 **Additional works in equivariant architectures for robot manipulation.** Aside from prior equiv-
 38 ariant architectures for robot manipulation introduced in the main paper, there are also robotics
 39 works that attempt to utilize equivariance in various setups. Some works [12, 13] attempt to use
 40 equivariance in a pick-and-place setup, while others [14, 15] propose $SO(2)$ -equivariant robot poli-
 41 cies for tabletop manipulation tasks. In comparison to prior works, our proposed architecture is
 42 equivariant to position, orientation, and uniform scaling. In addition, our method can be applied in
 43 various 3D manipulation tasks that involve rigid, deformable, and articulated objects.

44 B Limitations

45 B.1 Limitations of Our Method

46 In the paper, we mentioned the limitations of the method as well as the assumptions the method has
 47 to make about the input and output of the model. Here, we expand further on the limitations of our
 48 method.

49 While our method can generalize to scenes with unseen object positions, scales, and orientations, it
 50 does not generalize automatically to the following unseen scenarios: (1) there are multiple objects
 51 in the scene and their relative positioning varies at evaluation time; (2) object dynamics change (eg.
 52 cloth is more rigid at test time, thus resulting in a point cloud that has unseen shapes); (3) non-
 53 uniform scaling of objects; (4) unseen object shapes beyond simple scaling (eg. an unseen handle
 54 position of a cup). Although these limitations exist, it is important to mention two facts. First, our
 55 method strictly generalizes more than a vanilla Diffusion Policy, which our method is built on top
 56 of. Second, we can always collect demonstrations with varied object dynamics, relative positioning,
 57 and local shapes to make the resulting policy generalize to these aspects.

58 To make our method generalize out-of-distribution in the relative positioning of objects, object dy-
 59 namics, etc., future works can consider adding inductive biases in these aspects in the architecture
 60 of the model. For example, one can consider explicitly modeling multiple objects in the model
 61 to handle unseen relative positioning of different objects. One can also consider explicitly model-
 62 ing environment dynamics in an equivariant dynamics model to cope with unseen object dynamics.
 63 These directions are all interesting avenues for future work.

64 B.2 Failure Analysis

65 Although our method outperforms vanilla Diffusion Policy [4] and prior equivariant visuomotor
66 policy architectures, our method still presents various failure cases. Below, we focus our analysis
67 on execution failure of our method in the real robot experiments. In Figure 2, we show the failure
68 breakdown of all real robot executions we have performed with our method.

69 In most packing tasks (packing t-shirt, towel, and cap), the main failure cases are the end-effector
70 opening too early. We believe this is because the out-of-distribution scenarios resulted in the agent
71 thinking that it has moved to the dropping location for the object and opened the end-effector. For
72 the packing shorts task variation, half of the failures come from end-effector opening too late, and
73 half of the failures come from the shorts not slipping off from the end-effector due to the friction of
74 the end-effector. The failure to slip off happened because the shorts is a small deformable object that
75 easily hangs itself onto the end-effectors. This problem can potentially be solved by designing end-
76 effectors that can handle deformable objects better or performing online adaptation after training,
77 which is out of the scope of our work.

78 In the *push chair*, *laundry door closing*, and *bimanual folding* tasks, the majority of failures come
79 from the end-effector not performing the full motion or not performing gripper open close actions
80 at the right time. This most likely happens because the errors in predicted actions accumulated and
81 the observation became too out-of-distribution scenarios for the policy to behave correctly. In the
82 *bimanual make bed* task, our object segmentation algorithm appears to be more finicky than other
83 tasks, not segmenting the full comforter in some scenarios since the folded comforter looks like
84 two objects. This results in the policy sometimes not giving nice inference results, giving rise to
85 execution failures.

86 C Method Architectures and Implementation Detail

87 In this section, we describe in detail the architecture of our method. We visualize the architecture of
88 our model in Figure 3.

89 **Observation and action spaces.** In all simulated and real robot tasks except for *Push T*, we use a 13-
90 dimensional proprioception information and a 7-dimensional action space for each robot. The pro-
91 prioception data for each robot consists of the following information: a 3-dimensional end-effector
92 position, a 6-dimensional vector denoting end-effector orientation (represented by two columns of
93 the end-effector rotation matrix), a 3-dimensional vector indicating the direction of gravity, and a
94 scalar that represents the degree to which the gripper is opened. The action space for each robot
95 consists of the following information: a 3-dimensional vector for the end-effector position veloc-
96 ity, a 3-dimensional vector for the end-effector angular velocity in axis-angle format, and a scalar
97 denoting the gripper action.

98 In the *Push T* task, the robot proprioception is 3-dimensional and consists of the agent’s 3D position
99 in the scene, while the action space is 3-dimensional and denotes the absolute position target of the
100 agent.

101 In all simulated and real robot tasks, our policy uses an observation horizon of 2 steps, a prediction
102 horizon of 16 steps, and an action horizon of 8 steps. This is identical to the setup used in the
103 diffusion policy paper [4].

104 **Encoder architecture.** In all tasks except for *Push T*, we use a SIM(3)-equivariant version of
105 PointNet++ with 4 layers and hidden dimensionality 128. In the *Push T* task, we decrease the
106 number of layers to 2 since the number of points in the point cloud observation is much smaller in
107 this task.

108 **Noise prediction network.** Our noise prediction network inherits hyperparameters from the original
109 Diffusion Policy paper. In all simulation experiments, we use the DDPM scheduler [16] and perform
110 100 denoising steps during inference. In real robot experiments, to optimize for inference speed, we
111 use the DDIM scheduler [17] with 8 denoising steps.

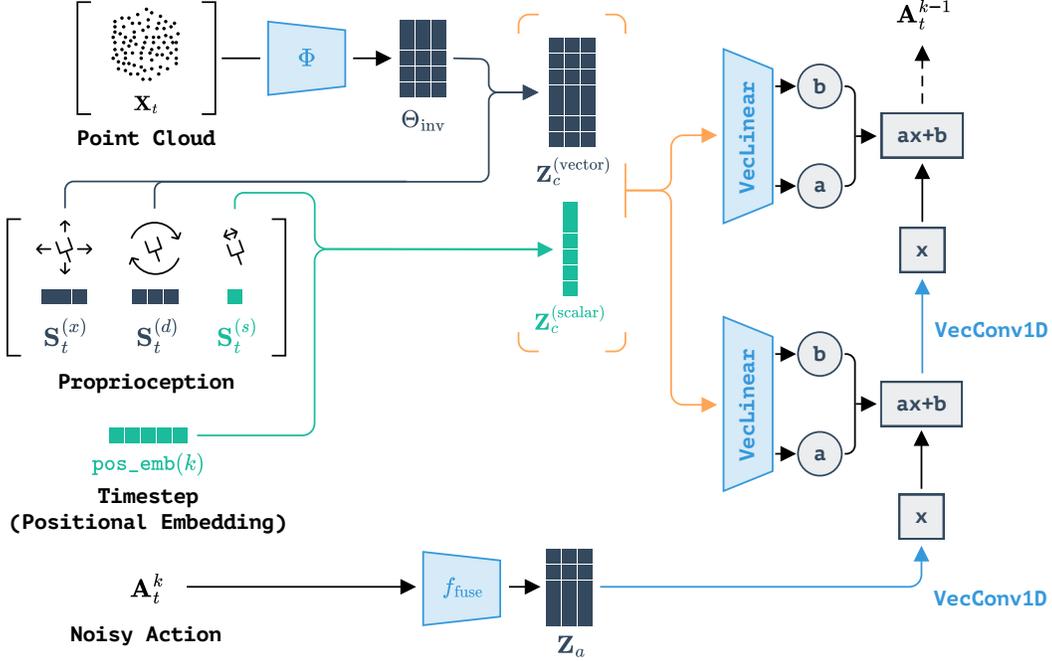


Figure 3: **Architecture of EquiBot.** Given input scene point cloud, robot proprioception, noisy actions, and the diffusion timestamp, our architecture processes position, direction, and scalar information independently, uses the encoder outputs to scale position information into position and scale invariant values, and then routes them into an SO(3)-equivariant conditional U-net to predict denoised actions. In the figure, we omit scaling for the ease of viewing. VecLinear and VecConv1D refer to a SO(3)-equivariant version of linear and convolution 1D layers.

112 **Point cloud size.** Picking the number of points to sample in the point cloud observation is a key
 113 hyperparameter to consider when designing an architecture that takes point cloud inputs. In our
 114 experiments, we found out that using 512 or 1024 points is sufficient for all tasks. In particular,
 115 for all real robot experiments and simulated mobile manipulation tasks, we use 1024-point point
 116 clouds. In *Can* and *Square* tasks, we use 256 and 512 points respectively since there is relatively
 117 more training data in these tasks, and decreasing the number of points in the point cloud makes
 118 training faster without hurting performance.

119 D Real Robot Setup Details

120 D.1 Human Demo Parsing Infrastructure

121 We use a single Zed 2 camera to record human natural motion in real-time, which is way more
 122 flexible and time-efficient than the expert demonstration from human teleoperation of a robot.

123 We assume access to a dataset $\mathcal{D} = \{\tau^n\}_{n=1}^N$ of human demonstrations. Each human demonstration
 124 consists of a series of RGB-D image frames $\tau^n = \{I_t^n\}_{t=1}^T$, where T is the episode horizon. The
 125 human demonstration processing module has three parts: (1) an off-the-shelf object detection and
 126 tracking model $X_t^n = \Psi(I_t^n, I_{t-1}^n)$ that takes the current and previous demonstration frames I_t^n
 127 and I_{t-1}^n (if existing) as input and outputs a parsed point cloud of objects of interest X_t^n ; (2)
 128 an off-the-shelf hand detection model $H_t^n = \Phi(I_t^n)$ that takes the current demonstration frame as input
 129 and outputs the keypoints on each human hand in the input frame H_t^n ; and (3) an alignment module
 130 $\Omega(X_t^n, H_t^n, I_t^n)$ that takes the outputs of the previous two steps as input, aligns the 3D coordinates
 131 of the outputs, and outputs the aligned human finger pose y_t^n in the same coordinate system of X_t^n .
 132 We use Grounded Segment Anything Model with DEVA [18] as the object detection and tracking
 133 model Ψ and HaMeR [19] as the hand detection model Φ . In the alignment module, we find a set of

134 matching points between the point cloud \mathbf{X}_t^n and \mathbf{H}_t^n , and then fit a rotation transformation R_h and
135 an offset t_h to transform all points \mathbf{H}_t^n to the coordinate frame of the point cloud. Then, we extract
136 the thumb and index finger positions on the transformed keypoint set to predict the human “end-
137 effector” pose \mathbf{y}_t^n . We found that the alignment module Ω is crucial, as the hand detection model
138 produces hand poses in a different coordinate frame from the point cloud. Without this module, the
139 resulting hand poses will not align with the object point cloud.

140 D.2 Mobile Robot Control Infrastructure

141 Our robot control setup consists of a centralized workstation and two mobile robots. The workstation
142 reads and parses visual observations, performs policy inference, and communicates with the robots.
143 The mobile robots take the output actions of the policy and execute them.

144 On the workstation side, we build a multi-processed infrastructure to handle observation parsing,
145 policy inference, and action execution separately. To reduce latency, we keep spinning the obser-
146 vation parsing and policy inference processes. We use shared memory to communicate between
147 different processes.

148 In the observation parsing process, we obtain visual observations from a single Zed 2 camera di-
149 rectly connected to the workstation via cable. We use the Grounded Segment Anything Model with
150 DEVA [18] to obtain segmented point clouds that contain only relevant objects in the scene. We
151 then downsample this segmented point cloud to 1024 points. The downsampled point cloud and
152 the robot proprioceptive information are sent to the policy inference process. The policy inference
153 process then outputs a sequence of 16 predicted actions.

154 In the action execution process, we first reset all robots to their initial poses. Then, for each step
155 in an evaluation episode, we read out the latest policy inference results from the policy inference
156 process. To ensure accurate execution, we compute the elapsed time between the policy input time
157 and action execution time. If this elapsed time exceeds a threshold, we skip the first few predicted
158 actions during action execution. After skipping the first few predicted actions to account for latency,
159 we select the 8 actions that immediately follow the skipped actions to execute. This means that no
160 matter how many actions are skipped, we always execute 8 actions at a time.

161 On the mobile robot, we also build a multi-processed control infrastructure and use shared memory
162 to communicate between different processes. There is a server process that captures all requests
163 from the clients and stores the command in shared memory to control the Kinova arm and mobile
164 base. As the arm and base operate at different frequencies, we decouple the control for those two
165 components but always ask the Kinova arm end-effector to track the expected global pose whenever
166 possible, no matter what the base pose is.

167 We utilize the mocap system in the room to update the mobile base position at 120 Hz. Then, we (1)
168 transfer each control signal from the global world frame to the local frame of the Kinova arm, (2)
169 convert the target pose at the gripper fingertip to an expected pose of the Kinova end-effector, and
170 (3) convert it into a velocity command for the arm. To ensure safe execution, a timeout callback will
171 stop the arm movement when no new velocity commands arrive for 4 times the expected execution
172 time interval. Unlike the control of the Kinova arm, we use position control for the mobile base and
173 will only move it when the robot end-effector is too close to or too far from the base. To ensure
174 safety, we define task-specific polygons workspace boundaries for the Kinova arm and the mobile
175 base separately, so that all motions are constrained within a safe region.

176 In future work, we will explore more principled whole-body control architecture to avoid decoupling
177 the arm and base motion.

178 D.3 Task Details

179 **Push chair.** In this task, the human demonstrations are collected using a standing desk (48×30
180 inches). The policies are evaluated on two different tables: a longer rectangular desk (58×23 inches)

Number of Demos →	Ours			DP		
	10	25	50	10	25	50
Success Rate	8/10	9/10	10/10	3/10	5/10	7/10
Close with Click Sound	2/10	2/10	5/10	2/10	1/10	5/10
Total Missing Angle	17.46°	7.18°	6.3°	140.42°	33.85°	21.55°
Collision or Safety Issue	0/10	0/10	0/10	2/10	0/10	0/10

Table 1: Detailed performance of the laundry door closing task.

181 and a circular table (diameter of 36 inches). An episode is considered successful if the center of the
 182 chair goes beneath the desk.

183 **Luggage packing.** In the human demonstrations, a human picks up a pack of white t-shirts and
 184 places them into a white carry-on luggage. At evaluation time, we test four different packing items:
 185 white t-shirts (same as training object), gray towel roll, blue cap, and navy shorts. An episode is
 186 considered successful if at least half of the packed object ends up within the luggage.

187 **Luggage closing.** In this task, human demonstrations are collected on a small carry-on luggage
 188 ($55 \times 40 \times 23$ cm), while the policies are evaluated on a large check-in luggage ($76 \times 48 \times 25$ cm).
 189 An episode is considered successful if the luggage ends up in a closed state.

190 **Laundry door closing.** In this task, the human demonstrations and the robot work with the same
 191 laundry machine (front-loader). The goal is to close the door of the laundry machine that is open at
 192 the start of the episode. An episode is considered successful if the door ends up with an opening of
 193 at most 5cm.

194 **Bimanual folding.** In this task, the human demonstrations are collected by using two hands to fold
 195 a small piece of cloth (34×38 cm). At evaluation time, the robot is asked to fold a large gray towel
 196 (140×75 cm). After each evaluation episode, we measure the mean distance between each grasped
 197 corner to their corresponding target cloth corners and mark the episode as successful if this mean
 198 distance is less than 0.2 times the length of the folding side of the cloth.

199 **Bimanual make bed.** In this task, the human demonstrations are collected by using two hands to
 200 unfold a towel (34×38 cm). At evaluation time, the robot is asked to make the bed by unfolding
 201 a much larger comforter on top of the bed. After each evaluation episode, we measure the mean
 202 distance between each grasped corner to the bed board and mark the episode as successful if this
 203 mean distance is less than 0.2 times the length of the bed.

204 E Additional Real Robot Results

205 **Detailed performance analysis of the laundry door closing task.** To understand the performance
 206 of the Diffusion Policy [4] and our method better, we perform a more detailed analysis in the laundry
 207 door closing task. In Table 1, we report four different metrics of policy performance in each evalua-
 208 tion setup. *Success Rate* measures the percentage of evaluations that end within the success criteria
 209 we set; *Close with Click Sound* measures the percentage of episodes that end with the laundry door
 210 closed completely after making a clicking sound; *Total Accumulated Missing Angle* measures the
 211 sum of the opening angles of the laundry door at the end of the 10 evaluation episodes; *Collision*
 212 *or Safety Issue* measures the percentage of evaluation runs that are terminated because of undesired
 213 collisions or critical safety issues, such as the robot arm getting stuck at the laundry door. From
 214 the evaluation results, we see that the *DP* policy not only has a lower success rate but also has a
 215 much larger accumulated missing angle. The baseline also suffers from many safety issues requir-
 216 ing episodes to be manually terminated by the robot operator. Our method has much fewer safety
 217 issues when it executes.

218 **Qualitative results.** In Figure 4, we show qualitative rollout samples for all evaluation scenarios
 219 we mentioned in the paper, plus one bonus task where two robots lift a woven basket onto a coffee
 220 table.



Figure 4: **Qualitative rollout samples for all real robot evaluation scenarios.** From top to bottom, we have: (1) pushing a chair towards a long standing desk; (2) pushing a chair towards a circular table; (3) packing t-shirts; (4) packing towel roll; (5) packing cap; (6) packing shorts; (7) closing a check-in luggage; (8) closing laundry door; (9) bimanual folding; (10) bimanual make bed; (11) a bonus task where two robots lift a woven basket onto a coffee table.

221 F Simulation Experiment Task Details

222 **Cloth folding.** In this task, the demonstrations show two robots folding a piece of cloth (27.5×27.5
 223 cm). During an evaluation, we compute the task reward as $1.0 - (d_1 + d_2)/(0.275 \times 2)$, where d_1
 224 and d_2 denote the distance from the two grasped cloth corners to the target cloth corners.

225 **Object covering.** In this task, the demonstrations show two robots moving a piece of cloth ($27.5 \times$
 226 27.5 cm) onto a rigid box ($10 \times 7 \times 5$ cm). During an evaluation, the task reward is computed as
 227 $V_{\text{intersect}}/V_{\text{convex hull}}$, where $V_{\text{intersect}}$ is the volume intersection between the box and the convex hull of
 228 the cloth and $V_{\text{convex hull}}$ is the volume of the convex hull of the cloth.

229 **Box closing.** In this task, the demonstrations show two robots closing a box ($14.5 \times 12 \times 11.5$ cm)
230 with three flaps. Success in this task is evaluated as $(a_1 + a_2 + a_3)/(3 \times 180)$, where a_1 to a_3 denote
231 the angle in degrees at which each flap of the box is closed.

232 **Push T.** In this task, a 2D anchor pushes a T shaped object on a plane of dimension 512×512 pixels.
233 The task reward is computed as the percentage of the T shape that overlaps with the target T pose.

234 **Robomimic tasks.** We use the same object and reward specifications in these tasks as the original
235 benchmark. Please check out the Robomimic [1] paper for more details.

References

- 236
- 237 [1] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese,
238 Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations
239 for robot manipulation. In *arXiv preprint arXiv:2108.03298*, 2021.
- 240 [2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Haus-
241 man, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv*
242 *preprint arXiv:2212.06817*, 2022.
- 243 [3] J. Yang, J. Zhang, C. Settle, A. Rai, R. Antonova, and J. Bohg. Learning periodic tasks from
244 human demonstrations. *IEEE International Conference on Robotics and Automation (ICRA)*,
245 2022.
- 246 [4] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy:
247 Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and*
248 *Systems (RSS)*, 2023.
- 249 [5] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu. 3d diffusion policy: Generalizable
250 visuomotor policy learning via simple 3d representations. In *Proceedings of Robotics: Science*
251 *and Systems (RSS)*, 2024.
- 252 [6] E. Hoogeboom, V. G. Satorras, C. Vignac, and M. Welling. Equivariant diffusion for molecule
253 generation in 3d. In *International conference on machine learning*, pages 8867–8887. PMLR,
254 2022.
- 255 [7] J. Guan, W. W. Qian, X. Peng, Y. Su, J. Peng, and J. Ma. 3d equivariant diffusion for target-
256 aware molecule generation and affinity prediction. *arXiv preprint arXiv:2303.03543*, 2023.
- 257 [8] I. Igashov, H. Stärk, C. Vignac, A. Schneuing, V. G. Satorras, P. Frossard, M. Welling, M. Bron-
258 stein, and B. Correia. Equivariant 3d-conditional diffusion model for molecular linker design.
259 *Nature Machine Intelligence*, pages 1–11, 2024.
- 260 [9] A. Schneuing, Y. Du, C. Harris, A. Jamasb, I. Igashov, W. Du, T. Blundell, P. Lió, C. Gomes,
261 M. Welling, et al. Structure-based drug design with equivariant diffusion models. *arXiv*
262 *preprint arXiv:2210.13695*, 2022.
- 263 [10] H. Ryu, J. Kim, J. Chang, H. S. Ahn, J. Seo, T. Kim, J. Choi, and R. Horowitz. Diffusion-edfs:
264 Bi-equivariant denoising generative modeling on $se(3)$ for visual robotic manipulation. *arXiv*
265 *preprint arXiv:2309.02685*, 2023.
- 266 [11] J. Brehmer, J. Bose, P. De Haan, and T. S. Cohen. Edgi: Equivariant diffusion for planning
267 with embodied agents. *Advances in Neural Information Processing Systems*, 36, 2024.
- 268 [12] H. Huang, D. Wang, R. Walters, and R. Platt. Equivariant transporter network. *arXiv preprint*
269 *arXiv:2202.09400*, 2022.
- 270 [13] H. Huang, O. Howell, X. Zhu, D. Wang, R. Walters, and R. Platt. Fourier transporter: Bi-
271 equivariant robotic manipulation in 3d. *arXiv preprint arXiv:2401.12046*, 2024.
- 272 [14] D. Wang, R. Walters, and R. Platt. $So(2)$ -equivariant reinforcement learning. *arXiv preprint*
273 *arXiv:2203.04439*, 2022.
- 274 [15] M. Jia, D. Wang, G. Su, D. Klee, X. Zhu, R. Walters, and R. Platt. Seil: Simulation-augmented
275 equivariant imitation learning. In *2023 IEEE International Conference on Robotics and Au-*
276 *tomation (ICRA)*, pages 1845–1851. IEEE, 2023.
- 277 [16] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in neural*
278 *information processing systems*, 33:6840–6851, 2020.

- 279 [17] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. *arXiv preprint*
280 *arXiv:2010.02502*, 2020.
- 281 [18] H. K. Cheng, S. W. Oh, B. Price, A. Schwing, and J.-Y. Lee. Tracking anything with decoupled
282 video segmentation. In *ICCV*, 2023.
- 283 [19] G. Pavlakos, D. Shan, I. Radosavovic, A. Kanazawa, D. Fouhey, and J. Malik. Reconstructing
284 hands in 3D with transformers. In *CVPR*, 2024.