

358 Appendix

359 6 DemoSpeedup pseudocode

360 We provide the complete pseudocode of *DemoSpeedup* in Algorithm 1.

Algorithm 1: *DemoSpeedup* for accelerating demonstrations to train visuomotor policy.

Input: $\mathcal{D} = \{\tau_i\}_{i=1}^B, \tau_i = \{o_t, a_t\}_{t=1}^T$; // original demonstrations
Train π_{proxy} **on** \mathcal{D} ; // train proxy policy on original demonstrations with ACT or DP
 // action entropy estimation via proxy policy
 def *get_entropy*(π_{proxy}, τ_i):
 Initialize: $t = 1, \mathcal{H}_{list} = [], S = \{\}, N$
 // \mathcal{H}_{list} is the entropy list, S is the action sample set, N is the number of samples
 while $t < T$ **do**
 for $i \leftarrow 1$ **to** N **do**
 $\hat{A}_t^i \leftarrow \pi_{proxy}(A_t | o_t, z_i)$; // A_t is the action chunk, z_i is sampled latent
 variable for ACT and sampled noise for DP
 Add action samples in \hat{A}_t^i to S ;
 end
 Calculate \mathcal{H}_t according to Equation 1, 2;
 Add \mathcal{H}_t to \mathcal{H}_{list} ;
 end
 return \mathcal{H}_{list} ;
 // accelerate demos with entropy
 def *accelerate_demos*($\mathcal{H}_{list}, \mathcal{D}$):
 Preprocess: $\mathcal{H}_{list} \leftarrow Isolation_Forest(\mathcal{H}_{list})$;
 Label Precision: $\{P, C\} \leftarrow Hdbscan_Cluster(\mathcal{H}_{list})$;
 Initialize: $\mathcal{D}_{speedup} = \{\tau_i^{speedup}\}_{i=1}^B, \tau_i^{speedup} = [], K$
 // K is the chunk size of accelerated policy
 for $i \leftarrow 1$ **to** B **do**
 Sample τ_i from \mathcal{D} ; **for** $t \leftarrow 1$ **to** T **do**
 Sample $\{o_t, a_{t:T}\}$ from τ_i ;
 $A_t^{speedup} \leftarrow piecewise_downsample_actions(a_{t:T}, \{P, C\})$;
 Add $\{o_t, A_t^{speedup}[:K]\}$ to $\tau_i^{speedup}$;
 end
 end
 return $\mathcal{D}_{speedup}$;
 // down-sample actions in sub-trajectories with precision label guidance
 def *piecewise_downsample_actions*($a_{t:T}, \{P, C\}$):
 Initialize: $r_{high}, r_{low}, indices = []$; // r_{high}, r_{low} is high and low down-sample ratio
 for $i \leftarrow t$ **to** T **do**
 if $[i : i + r_{high}] \subseteq C$ **then**
 $i \leftarrow i + r_{high}$, Add i to $indices$;
 else
 $i \leftarrow i + r_{low}$, Add i to $indices$;
 end
 end
 $A_t^{speedup} \leftarrow a_{indices}$;
 return $A_t^{speedup}$;
Train $\pi_{speedup}$ **on** $\mathcal{D}_{speedup}$ **by directly imitating** $\{o_t, A_t^{speedup}[:K]\}$;
 // train accelerated policy on accelerated demonstrations with ACT or DP
output: $\pi_{speedup}$

7 Additional Comparisons

7.1 Comparison with other demonstration speedup methods

We further compare the performance of *DemoSpeedup* with other down-sample baselines by replacing the entropy guided piecewise acceleration with following methods:

- *Contact Oracle*: We design a contact-based heuristic to segment low-precision and high-precision subtrajectories. The division rule is as follows: whenever a new object-pair contact or detachment occurs in the Manipulation scene (e.g., the gripper contacts with the object, or one object contacts with another object), a constant time before and after the moment when the contact state changes are labeled as precision. The rest of the time is labeled as casualness. Note that this method requires oracle information and precise 3D priors, which are difficult to obtain in real-world settings with only 2D camera inputs.
- *AWE**: We adjust a dynamic programming method from AWE[27] to down-sample the data. AWE aims to promote success rate. It minimizes the trajectory length by $7\times-10\times$ given a threshold constraint of piece-wise linear approximation error of the joint-angle trajectories. AWE needs much longer time than demonstrations to reach waypoints. So we tune the threshold to reduce the trajectory length to roughly $2\times$ and use the same control frequency as demonstrations for acceleration. Besides, AWE only relies on joint-angle trajectories which can be noisy and task-irrelevant. We improve it by re-weighting the approximation error with entropy.
- *Constant $2\times$* : Directly down-sample the demonstrations at $2\times$ ratio.
- *Constant $3\times$* : Directly down-sample the demonstrations at $3\times$ ratio.

Method & Algo	Transfer Cube&ACT		Insertion&ACT		Transfer Cube&DP		Insertion&DP	
	success rate (\uparrow)	episode len (\downarrow)	success rate (\uparrow)	episode len (\downarrow)	success rate (\uparrow)	episode len (\downarrow)	success rate (\uparrow)	episode len (\downarrow)
<i>Origin</i>	40%	321	11%	435	47%	289	12%	329
<i>Contact Oracle</i>	37%	140	15%	142	37%	124	11%	127
<i>DemoSpeedup</i>	40%	137	22%	125	49%	121	16%	145

Table 4: **Comparison with *Contact Oracle***. We collect a new dataset including contact information using a trained checkpoint to conduct the experiment. *DemoSpeedup* achieves a comparable success rate with *Origin* while *Contact Oracle* often performs worse than *Origin*.

Method & Algo	Transfer Cube&ACT		Insertion&ACT		Transfer Cube&DP		Insertion&DP	
	success rate (\uparrow)	episode len (\downarrow)	success rate (\uparrow)	episode len (\downarrow)	success rate (\uparrow)	episode len (\downarrow)	success rate (\uparrow)	episode len (\downarrow)
<i>Origin</i>	72%	291	21%	452	66%	281	16%	431
<i>AWE*</i>	63%	148	14%	183	53%	169	9%	221
<i>Constant $2\times$</i>	80%	167	27%	242	75%	152	20%	247
<i>Constant $3\times$</i>	47%	126	7%	163	39%	109	4%	198
<i>DemoSpeedup</i>	81%	121	30%	151	74%	107	29%	218

Table 5: **Comparison with other baselines**. Compared to other down-sample methods, our approach achieves the best balance between success rate and speed.

Additionally, we utilize *Origin* to refer policies trained on original demonstrations. Other factors including replicate-before-downsample and Geometrical consistency are the same to guarantee a fair comparison. We conduct experiments on Aloha with ACT and DP. To compare with *Contact Oracle*, since the original datasets doesn’t offer any privileged information, we recollect 50 new demonstrations including the contact information by rollouting a trained ACT. Then *Contact Oracle*, *Origin* and *DemoSpeedup* are all trained on this dataset for comparison. To compare with other methods, we still use the original dataset.

The comparison results with *Contact Oracle* are shown in Table 4. *DemoSpeedup* achieves a better performance than *Contact Oracle* while achieving similar speedup. *Contact Oracles* often falls short of the success rate of *Origin*. We argue that this is because contact pattern can’t account for all precision patterns and could only offer a rough estimation of precision. For example, in Insertion task, the contact pattern keeps the same after one block first contacts with the other block, thus failing to capture the contact-rich phase of one block being inserted to the other. Besides, the constant time period around the contact change moment fails to offer an accurate estimation of precision stage duration.

The comparison results with other methods are shown in Table 5. *DemoSpeedup* strikes the best balance between the success rate and speedup. Specifically, *DemoSpeedup* achieves a similar performance with *Constant 2×* and a similar speedup with *Constant 3×*. Additionally, though AWE could promote success rate in its original setting, we find AWE* even worse than *Constant 2×* for acceleration. It is mostly because the approach is based on dynamic programming which doesn’t consider the smoothness of selected actions. Therefore, the accelerated policy often produces jittery motions, which hurts the performance. This demonstrates the unique challenge of accelerating policy execution that is different from traditional down-sample settings.

7.2 Comparison with traditional down-sample approaches

Down-sampling the demonstrations has been a widely used practice in robot community. However, most down-sample approaches serve for improving the performance rather than accelerating policy execution. *DemoSpeedup* differs from previous approaches in following two ways:

- **Execution frequency.** Our down-sample method decreases the demonstration frequency but doesn’t decrease the policy execution frequency. For example, for a 50Hz recorded demonstration, traditional methods may down-sample it to 20Hz and deploy the trained policy at 20Hz. However, in our setting, we down-sample it at 20Hz but deploy the checkpoint at original 50Hz in order to accelerate execution.
- **Novel Challenge.** The main challenge in this work is to maintain the performance while speeding up the execution. Thus even a $< 5\%$ drop in success rate is intolerable. Besides, previous methods such as Keyframes[42] rely on close-loop control to reach down-sampled waypoints, so the speed is even slower than original demonstrations. On the contrary, the execution speed in this work is much faster than the demonstrations. Thus, the challenge of acceleration demands a much higher accuracy in recognizing precision patterns than traditional settings. Traditional heuristic methods perform poorly against this challenge, as shown in 7.1. *DemoSpeedup* well mitigates this challenge using entropy and maintains the performance.

8 Hyperparameters

8.1 High and low down-sample ratio

Task	$\{r_{low}, r_{high}\}$	Task	$\{r_{low}, r_{high}\}$
Transfer Cube	$\{2, 4\}$	Open Trays	$\{2, 4\}$
Insertion	$\{2, 4\}$	Flip Cutlery	$\{1, 3\}$
Sandwich Remove	$\{2, 4\}$	Cupboard Open	$\{2, 4\}$
Move Plate	$\{2, 4\}$	Pen in Cup	$\{2, 4\}$
Load Cups	$\{2, 4\}$	Sort	$\{2, 4\}$
Put Cups	$\{2, 4\}$	Kitchenware	$\{2, 4\}$
Saucepan to Hob	$\{2, 4\}$	Bomb Deposal	$\{2, 3\}$
Drawer Close	$\{2, 4\}$	Conveyer	$\{2, 4\}$

Table 6: Hyperparameter of high and low down-sample rate for ACT.

The key hyperparameters in *DemoSpeedup* is r_{high} and r_{low} , the high and low down-sample ratio. They directly impact the performance and speedup of accelerated policy. We keep them the same

Task	$\{r_{low}, r_{high}\}$	Task	$\{r_{low}, r_{high}\}$
Transfer Cube	$\{2, 4\}$	Open Trays	$\{2, 4\}$
Insertion	$\{2, 4\}$	Flip Cutlery	$\{1, 3\}$
Sandwich Remove	$\{2, 4\}$	Cupboard Open	$\{2, 4\}$
Move Plate	$\{2, 3\}$	Pen in Cup	$\{2, 4\}$
Load Cups	$\{2, 4\}$	Sort	$\{2, 4\}$
Put Cups	$\{2, 3\}$	Kitchenware	$\{2, 3\}$
Saucepan to Hob	$\{2, 4\}$	Bomb Deposal	$\{2, 3\}$
Drawer Close	$\{2, 4\}$	Conveyer	$\{2, 4\}$

Table 7: Hyperparameter of high and low down-sample rate for DP.

in most tasks, proving the robustness and generalization of our approach. However, if some non-accelerated demonstrations are fast enough or the task requires extremely precision, we need to manually tune them via several trials. r_{high} and r_{low} are shown in Table 6 and 7. Empirically, selecting down-sample rate is relatively easy, as we only use integers as the down-sample ratio. One can train checkpoints for multiple ratios simultaneously and evaluate which works best.

8.2 ACT Hyperparameters

We utilize the same hyperparameter configuration for ACT across all tasks, shown in Table 8. For chunk length, we utilize time duration to represent it, as different tasks in this work have different control frequencies. The specific chunk size is the chunk length multiplied by the control frequency. For ACT+*DemoSpeedup*, we use half of ACT’s chunk length to ensure geometrical consistency. This is based on the observation that most speedups in our experiments are around $2\times$.

Hyperparameter	ACT	ACT + <i>DemoSpeedup</i>
learning rate	1e-5	1e-5
# encoder layers	4	4
# decoder layers	7	7
feedforward dimension	3200	3200
hidden dimension	512	512
# heads	8	8
chunk length	1s	0.5s
beta	10	10
dropout	0.1	0.1

Table 8: Hyperparameters of ACT + *DemoSpeedup* and ACT. The only difference is the reduction in chunk size.

Hyperparameter	Aloha	Bigym	Real-world
observation horizon	1	1	1
diffusion_step_embed_dim	128	256	128
down_dims	[512,1024,2048]	[256,512,1024]	[512,1024,2048]
kernel_size	5	5	5
n_groups	8	8	8
vision_model	Resnet18	MVT[36]	Resnet18
chunk size	48	16	24
Lr	1e-4	1e-4	1e-4
WDecay	1e-6	1e-6	1e-6
scheduler	DDIM	DDIM	DDIM
D -Iters Train	100	100	100
D -Iters Eval	10	10	10

Table 9: Hyperparameters for Diffusion Policy.

Hyperparameter	Aloha	Bigym	Real-world
observation horizon	1	1	1
diffusion_step_embed_dim	128	256	128
down_dims	[512,1024,2048]	[256,512,1024]	[512,1024,2048]
kernel_size	5	5	5
n_groups	8	8	8
vision_model	Resnet18	MVT[36]	Resnet18
chunk size	24	8	12
Lr	1e-4	1e-4	1e-4
WDecay	1e-6	1e-6	1e-6
scheduler	DDIM	DDIM	DDIM
D -Iters Train	100	100	100
D -Iters Eval	10	10	10

Table 10: **Hyperparameters for DP+*DemoSpeedup***. The only difference with DP is the reduction in chunk size.

8.3 Diffusion Policy Hyperparameters

We utilize separate hyperparameter configurations across Aloha, Bigym and real world for best performance, shown in Table 9, 10. All the DP used in our experiments are CNN-based.