SwiftPolicy: Accelerated Visuomotor Policies via Score Distillation Sampling

Duc-Hai Pham Ianis Hammani
ENS Paris-Saclay

Abstract

For a robot to work robustly in the real world, a policy learning protocol must be able to handle multi-modal action distributions and high-dimensional sensory data. Diffusion Policy (DP) addresses this by treating policy learning as a conditional denoising diffusion process, bypassing the instabilities of traditional energy-based. However, the iterative nature of diffusion models introduces a latency bottleneck, posing a challenge for deployment on low-resource hardware. In this report, we propose SwiftPolicy, a model distillation scheme designed to compress the Diffusion Policy into a reactive, one-step student network. Inspired by advancements in text-to-image distillation, SwiftPolicy utilizes Score Distillation Sampling (SDS) to guide the student model in directly translating noise to action without iterative refinement. Our evaluations in toy simulation benchmarks demonstrate that the one-step student achieves task completeness and success rates comparable to the multi-step teacher. This opens new avenues for deploying DPs on consumer-grade robotic platforms. ***Our code is released****: github.com/haiphamcse/SwiftPolicy*

1. Introduction

Behavior Cloning (BC) has emerged as a compelling paradigm for endowing robots with complex motor skills by learning directly from human demonstrations. However, real-world human movements present a fundamental challenge: multimodality. For any given observed state, there are often multiple valid strategies to complete a task (e.g., navigating around an obstacle to the left or the right).

Previous works in policy learning like Gaussian Mixture Models [5] and categorical distribution [8] often struggle with high-dimensional action spaces and limited modeling capacity.

Diffusion Policy (DP) [1] recently bridged this gap by modeling the policy as a conditional denoising diffusion process. By learning the gradient of the score function, DP captures multimodal distributions with the training stability of regression, establishing a new state-of-the-art in robotic manipulation. Despite this performance, DP suffers from a critical bottleneck: inference latency. The formulation of Denoising Diffusion Probabilistic Models (DDPMs) [3] necessitates an iterative denoising process, often requiring tens or hundreds of network evaluations to generate a single action.

This limitation motivates the search for a "one-step" solution. In the domain of text-to-image generation, recent works [6, 11] have successfully distilled the complex mappings of multi-step diffusion teachers into single-pass student networks. We ask: **can similar distillation techniques be adapted to accelerate robotic policies?**

In this work, we propose SwiftPolicy, a model distillation scheme that compresses a pre-trained Diffusion Policy into a reactive, one-step student network. To overcome the challenge of matching the teacher's trajectory without costly simulation, we leverage Score Distillation Sampling (SDS) [7]. We utilize the teacher's denoised outputs not as ground truth targets, but as gradient directions. The SDS loss guides the student by providing implicit gradients that nudge its output toward the high-density regions of the teacher's distribution, effectively teaching the student to generate valid actions in a single forward pass.

We evaluate SwiftPolicy on the PushT simulation benchmark [2] using both vision and state-based conditioning. Our results demonstrate that the one-step student achieves task completion (coverage) rates comparable to the multi-step teacher. Remarkably, the student achieves this robustness without requiring additional rollouts, indicating that it successfully captures the underlying action manifold.

Our contributions are summarized as follows:

- We introduce SwiftPolicy, a distillation framework that transforms a multi-step Diffusion Policy into a fast, one-step policy network.
- We propose the use of Score Distillation Sampling for robotic control, utilizing score gradients to implicitly map the teacher's expressive distribution to the student.
- We demonstrate through simulation that SwiftPolicy matches the action quality and success rates of the teacher model while achieving a $50\times$ reduction in neural network evaluations.

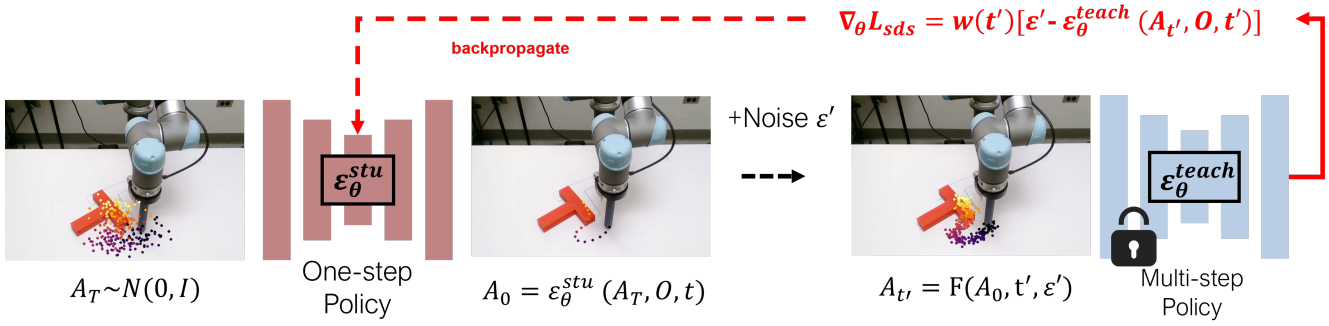

Figure 1. **Method Overview.** We train a reactive one-step student policy ϵ_θ^{stu} to directly map random noise A_T into a clean action trajectory A_0, conditioned on observation O. To align the student with the multi-step teacher's distribution, we employ *Score Distillation Sampling* (SDS). The student's prediction is perturbed with noise ϵ' to a random timestep t', and the frozen teacher ϵ_θ^{teach} predicts the noise residual. The difference between the injected noise and the teacher's prediction drives the gradient update $\nabla_\theta \mathcal{L}_{SDS}$ (red dashed line), guiding the student to generate valid actions in a single forward pass.

2. Methodology

In this section, we present the theoretical underpinnings and practical implementation of SwiftPolicy. We begin with a review of Diffusion Policy (DP) and Score Distillation Sampling (SDS). We then detail our primary contribution: a distillation protocol that compresses a multi-step teacher policy into a reactive one-step student using SDS. Finally, we provide the specific algorithmic implementation.

2.1. Preliminaries

Diffusion Policy. The method adapts *Denoising Diffusion Probabilistic Models* (DDPM) [3] to the visuomotor context. Let $\mathbf{A}^k$ be the sequence of actions at diffusion step k. The forward process diffuses information by progressively adding Gaussian noise over K iterations, transforming the data distribution $p_{data}(\mathbf{A}^0)$ into isotropic white noise $\mathcal{N}(\mathbf{0}, \mathbf{I})$.

The objective is to **learn the reverse process** parameterized by θ, which reconstructs the clean action $\mathbf{A}^0$ from noise $\mathbf{A}^K$ conditioned by observation $\mathbf{O}$. Generation is governed by **Stochastic Langevin Dynamics**. The update to pass from step k to $k-1$ is written as:

$$\mathbf{A}^{k-1} = \alpha\left(\mathbf{A}^k - \gamma\epsilon_\theta(\mathbf{O}, \mathbf{A}^k, k) + \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})\right) \quad (1)$$

where ϵ_θ is the noise prediction network, and α, γ, σ are scalar coefficients defined by the noise schedule. The network is trained to approximate the noise ϵ added to the clean action via a Mean Squared Error (MSE) objective:

$$\mathcal{L}_{DP} = \mathbb{E}_{k,\mathbf{A}^0,\epsilon}\left[||\epsilon - \epsilon_\theta(\mathbf{O}, \mathbf{A}^k, k)||^2\right] \quad (2)$$

Minimizing this objective is equivalent to *Denoising Score Matching* [9]. The network learns to approximate the gradient of the score function of the data distribution, $\epsilon_\theta(\mathbf{O}, \mathbf{A}) \propto -\nabla_{\mathbf{A}} \log p(\mathbf{A}|\mathbf{O})$, effectively learning the energy field without estimating the intractable normalization constant Z.

Unlike classical policies that predict a single action a_t, Diffusion Policy predicts a sequence of future actions $A_t = [a_t, \ldots, a_{t+T_p}]$. Execution follows a predictive scheme:

- At each time step t, the model predicts a trajectory of length T_p (*Prediction Horizon*).
- The robot executes a sub-sequence of length T_a (*Action Horizon*), with $T_a < T_p$.
- At $t + T_a$, the process restarts.

The authors [1] demonstrate that this temporal redundancy is crucial for movement smoothness and robustness to computational latencies. A typical observed value is $T_p = 16$ and $T_a = 8$.

Score Distillation Sampling (SDS). SDS is a technique originally developed for text-to-3D generation [4, 7, 10]. It utilizes a pre-trained diffusion model to optimize a differentiable parametric representation (e.g., a NeRF parameterized by θ).

Given a differentiable rendering function $g(\cdot, c)$ and camera parameters c, the SDS loss updates θ such that the rendered image $x = g(\theta, c)$ aligns with the distribution of the diffusion model. The gradient of the SDS loss is approximated as:

$$\nabla_\theta \mathcal{L}_{SDS} = \mathbb{E}_{t,\epsilon,c}\left[w(t)\left(\epsilon_\psi(x_t, t, y) - \epsilon\right)\frac{\partial g(\theta, c)}{\partial \theta}\right] \quad (3)$$

where ϵ_ψ is the frozen teacher, x_t is the noisy rendering, and $w(t)$ is a weighting function. Our insight is that this formulation can be generalized: we replace the "renderer" with a student policy and the "image" with a robot action sequence.

2.2. SwiftPolicy Distillation

Framework. Our framework consists of two models: a frozen multi-step teacher ϵ_θ^{teach} and a trainable one-step student ϵ_ϕ^{stu}. Our objective is to refine the student so that it predicts a clean, noise-free action $\mathbf{A}_0$ in a single forward pass, mimicking the distribution learned by the teacher.

As illustrated in Fig. 1, the distillation process avoids the computational cost of full teacher rollouts. Instead, it relies on score-based guidance:

1. **Student Prediction:** We sample Gaussian noise $\mathbf{A}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and condition the student on the current observation $\mathbf{O}$. The student predicts a clean action candidate $\hat{\mathbf{A}}_0 = \epsilon_\phi^{stu}(\mathbf{A}_T, \mathbf{O}, T)$.
2. **Diffusion (Re-noising):** To query the teacher's knowledge, we must map this prediction back to a noisy manifold. We sample a random timestep $t \sim \mathcal{U}(0, T)$ and add noise ϵ' to obtain $\mathbf{A}_t$.
3. **Teacher Critique:** The teacher predicts the noise residual $\hat{\epsilon}_{teach} = \epsilon_\theta^{teach}(\mathbf{A}_t, \mathbf{O}, t)$.
4. **Gradient Update:** We compute the SDS gradient to update the student parameters ϕ. The gradient pushes the student's output such that, when re-noised, it matches the noise predicted by the teacher.

Adapting Eq. 3 to our control setting, the gradient update rule becomes:

$$\nabla_\phi \mathcal{L}_{SDS} = w(t) \left(\epsilon_\theta^{teach}(\mathbf{A}_t, \mathbf{O}, t) - \epsilon'\right) \frac{\partial \hat{\mathbf{A}}_0}{\partial \phi} \quad (4)$$

This allows the student to learn the multimodal action distribution captured by the teacher without requiring explicit ground-truth targets or expensive iterative sampling during training.

2.3. Implementation

We detail the training procedure in Algorithm 1. We utilize a weighting function $w(t) = 1 - \bar{\alpha}_t$ to stabilize gradients across different noise levels. A crucial implementation trick here is that even-though the student is parameterized as ϵ-prediction, we directly convert it to x_0-prediction without any reparameterization. This improves stability during training/inference, although we have no idea why it is the case (may need further research on this).

Algorithm 1 SwiftPolicy: One-Step SDS Distillation

Input: Pre-trained Teacher ϵ_θ, Student ϵ_ϕ, Dataset $\mathcal{D}$, Noise Schedule $\{\bar{\alpha}_t\}_{t=1}^T$
Initialize: Student weights $\phi \leftarrow \theta$ (warm start)
for iteration $k = 1$ **to** K **do**
  for batch $(\mathbf{O}, \dots) \sim \mathcal{D}$ **do**
    Sample initial noise $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    1. Student Prediction:
    $\hat{\mathbf{A}}_0 \leftarrow \epsilon_\phi(\mathbf{z}, \mathbf{O}, T)$
    2. Re-noising Process:
    Sample $t \sim \mathcal{U}(0, T)$ and $\epsilon' \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
    $\mathbf{A}_t \leftarrow \sqrt{\bar{\alpha}_t}\hat{\mathbf{A}}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon'$
    3. Teacher Prediction:
    $\hat{\epsilon}_{teach} \leftarrow$ no_grad$(\epsilon_\theta(\mathbf{A}_t, \mathbf{O}, t))$
    4. SDS Update:
    $w(t) \leftarrow (1 - \bar{\alpha}_t)$
    $\nabla_\phi \mathcal{L}_{\mathcal{SDS}} = w(t)(\hat{\epsilon}_{teach} - \epsilon') \frac{\partial \hat{\mathbf{A}}_0}{\partial \phi}$
    Update ϕ using Adam optimizer
  end for
end for

3. Experiments

3.1. Setup

Evaluation dataset and metric. We evaluate our method on the *Push-T* task adapted from Implicit Behavioral Cloning (IBC) [2]. This task involves pushing a T-shaped block (gray) to a fixed target pose (red) using a circular end-effector (blue). The environment initializes with random positions for both the block and the end-effector, requiring the agent to exploit contact-rich object dynamics to achieve precise alignment. We evaluate two variants: *Vision-based* (using RGB image observations) and *State-based* (using 9 2D keypoints derived from the block's ground-truth pose). Both variants utilize proprioception for the end-effector location.

To quantify performance, we utilize *target coverage*. Following Diffusion Policy, we compute a coverage ratio C representing the geometric overlap between the object and the target. Let $\mathcal{B} \subset \mathbb{R}^2$ be the block geometry and $\mathcal{G} \subset \mathbb{R}^2$ be the goal geometry. The coverage is defined as:

$$C = \frac{\text{Area}(\mathcal{B} \cap \mathcal{G})}{\text{Area}(\mathcal{G})} \quad (5)$$

where the area is calculated via Shapely polygons. This metric yields a scalar $C \in [0, 1]$, where $C > 0.95$ typically indicates a successful task completion.

Implementation Details. We utilize the official multi-step Diffusion Policy checkpoints provided by the authors as our teacher models. To reduce computational costs, we employ the CNN-based U-Net architecture as the noise predictor, though our distillation framework is architecture-agnostic (e.g., compatible with Transformers). We freeze the teacher weights during distillation. The student model is trained for 20 epochs on a single NVIDIA T4 GPU.

During inference, we employ Receding Horizon Control (RHC) with a prediction horizon $T_p = 16$ and an action ex-

ecution horizon $T_a = 8$. For baselines, we compare against the Teacher utilizing both the DDPM (50 steps) and DDIM (10 steps) sampling schedulers. We run all evaluations on 50 episodes with a maximum rollout of 200 and different seeds for each episode.

3.2. Quantitative Results

We first validate our setup by reproducing the baseline Diffusion Policy results. As shown in Table 1, our reproduction yields an average coverage of $\sim 70\%$. While slightly lower than the reported 84%, likely due to differences in evaluation seed variance, the model retains high stability and distinct multimodal behaviors, serving as a valid teacher for distillation.

Metric	Reported [1]	Reproduced (Ours)
Average Coverage	84%	$\sim 70\%$
Training Stability	High	High
Multimodal Behavior	Distinct Modes	Distinct Modes

Table 1. Verification of the teacher baseline on the vision-based Push-T task.

Distillation Performance. Table 2 summarizes the performance of the distilled Student against the Teacher. We report both the *Maximum Coverage* (best episode performance) and the *Mean Coverage* across all evaluation rollouts.

Model	NFE ↓	Vision-based Max / Mean Cov.	State-based Max / Mean Cov.
DDPM	50	1.00 / 0.657	1.00 / 0.891
DDIM	10	1.00 / 0.621	1.00 / 0.893
SwiftPolicy	**1**	**1.00** / 0.537	**1.00** / 0.710

Table 2. Performance comparison. **NFE** denotes Neural Function Evaluations per action inference. The Student achieves a **50× speedup** and maintains the ability to solve the task (Max Cov. 1.0), albeit with a moderate drop in average consistency.

The results highlight a significant efficiency gain: the Student requires only **1 Neural Function Evaluation (NFE)** compared to the Teacher's 50, drastically reducing latency. In terms of task success, the Student retains the capacity to solve the task perfectly (Max Coverage of 1.0 in both domains). However, we observe a degradation in *average* coverage ($\sim 12\%$ drop in vision, $\sim 18\%$ drop in state-based). This indicates that while the Student can perform optimal actions, it is slightly less robust to initial condition variance than the iterative Teacher.

3.3. Qualitative Analysis

Mode Preservation. A critical risk in distilling generative models into regression policies is "mode collapse," where the model averages conflicting actions. We visualize sample trajectories in Figure 2. The Student (red) clearly preserves the **multimodality** of the Teacher (blue), choosing distinct paths around obstacles rather than averaging them. This confirms that the SDS loss successfully transfers the structural properties of the diffusion distribution.

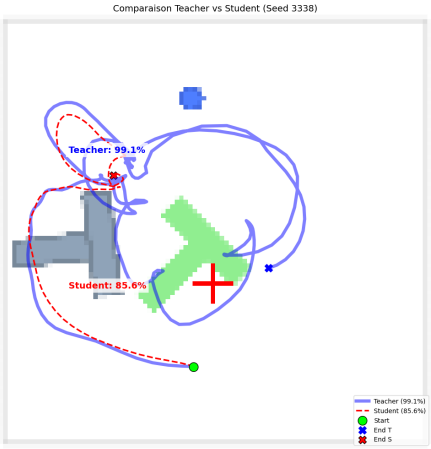

Figure 2. Trajectory Analysis. The One-Step Student (red) maintains the multimodal decision-making of the Teacher (blue), navigating around the target rather than averaging the path.

Behavioral Complexity and Noise. The quality of the one-step transfer appears sensitive to initial conditions.

1. **Complex Scenarios:** When the optimal trajectory requires long-horizon planning or complex manipulation, the Student exhibits noisier behavior (Figure 3). While it avoids mode collapse, the resulting trajectory is less smooth than the Teacher's. We hypothesize this is due to the "distillation intensity", i.e a single SDS step may not fully resolve the fine-grained details of the score function in low-density regions.
2. **Edge Case Failures:** In extreme initial states (e.g., block far from target), the Student may fail to reach the success threshold (Figure 4). This is partly inherited from the Teacher; if the Teacher's probability mass is spread thin in these regions, the SDS gradients become noisy, leading to suboptimal student convergence.
3. **Trajectory Smoothing:** Conversely, for simpler initial conditions, the Student occasionally outperforms the Teacher by producing **more straightforward trajectories** (Figure 5). By condensing the iterative process, the Student can bypass some of the stochastic "jitter" inherent in the multi-step denoising process, resulting in cleaner execution for short-horizon sub-tasks.

4. Conclusion

We explored the *Diffusion Policy* framework, a paradigm shift in robot learning that treats policy learning as a con-

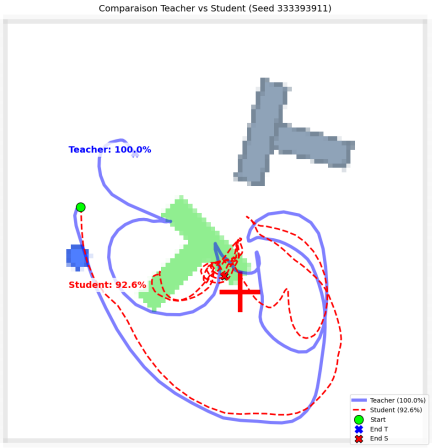

Figure 3. Noisy Student behavior in complex states.

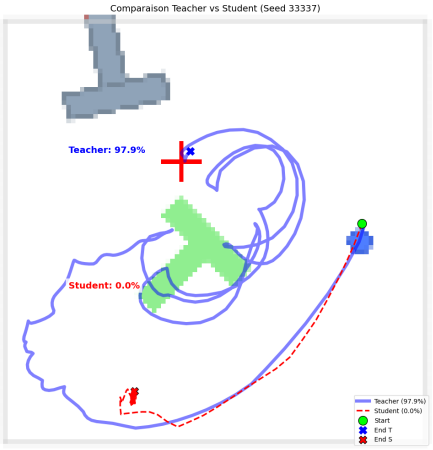

Figure 4. Failure mode in extreme initial conditions.

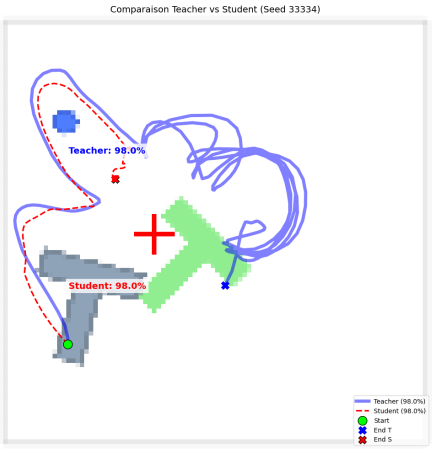

Figure 5. Unexpected Benefit: The Student produces a more direct trajectory than the Teacher in simple configurations.

ditional generative modeling problem. Theoretical analysis highlighted how the score-matching objective circumvents the instability of Energy-Based Models while capturing the multimodality inherent in human demonstrations.

Our reproduction of the results confirmed the high performance of the method but also validated the concern regarding inference latency, which creates a bottleneck for real-time reactivity.

To address this, we implemented a **Score Distillation Sampling (SDS)** protocol on two benchmarks. Our experimental results demonstrate that it is possible to distill the complex diffusion distribution into a fast, one-step student policy. The student achieve a $\times 50$ speedup, suitable for real-time control on a robot, with a manageable trade-off in success. This suggests that Diffusion Policies can serve not only as high-performance controllers but also as effective "Teachers" for training lightweight, reactive policies deployable on hardware with limited compute.

References

[1] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44 (10-11):1684–1704, 2025.

[2] Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Conference on robot learning*, pages 158–168. PMLR, 2022.

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[4] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 300–309, 2023.

[5] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[6] Thuan Hoang Nguyen and Anh Tran. Swiftbrush: One-step text-to-image diffusion model with variational score distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7807–7816, 2024.

[7] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[8] Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. *Advances in neural information processing systems*, 35:22955–22968, 2022.

[9] Pascal Vincent. A connection between score matching and

denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.

[10] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12619–12629, 2023.

[11] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024.