
Drag-guided diffusion models for vehicle image generation

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 Denoising diffusion models trained at web-scale have revolutionized image genera-
2 tion. The application of these tools to engineering design holds promising potential
3 but is currently limited by their inability to understand and adhere to concrete
4 engineering constraints. In this paper, we take a step toward the goal of incorpo-
5 rating quantitative constraints into diffusion models by proposing physics-based
6 guidance, which enables the optimization of a performance metric (as predicted
7 by a surrogate model) during the generation process. As a proof-of-concept, we
8 add drag guidance to Stable Diffusion, which allows this tool to generate images of
9 novel vehicles while simultaneously minimizing their predicted drag coefficients.

10 1 Introduction

11 Diffusion models have produced state-of-the-art results for high-resolution image generation. One
12 powerful aspect of diffusion models is the incorporation of *text-based guidance*, which steers the
13 image generation process towards an output that matches a user-provided description. Such guidance
14 is implemented in DALL-E 2 [25], Imagen [30] and Stable Diffusion [27]. Models trained on image-
15 caption pairs from the internet, however, do not provide the capability to specify precise quantitative
16 engineering constraints through text guidance.

17 To successfully incorporate engineering constraints into existing image generation tools, new guidance
18 approaches must be developed. In this paper, we implement *physics-based* guidance that simultane-
19 ously estimates and optimizes the performance of the generated object. As a proof-of-concept, we
20 implement *drag* optimization of *vehicles*, but our approach can be adapted to any design task that
21 admits a *surrogate model* of performance with the following characteristics: (1) the input is an image,
22 as produced by a diffusion process, (2) the model is robust to distributional shifts in the input, and (3)
23 the model is differentiable, allowing for gradient-based optimization of the output.

24 For the task of vehicle drag optimization, we train a surrogate model that meets these specifications
25 using the dataset [31], which consists of two-dimensional (2D) vehicle renderings labeled with drag
26 coefficients calculated from computational fluid dynamics (CFD) simulations. Aiming for robustness
27 to distributional shifts from our training set, we build our model using feature-space embeddings from
28 widely used object recognition networks [23, 12, 9]. Our surrogate model is naturally differentiable
29 because it is implemented as a deep neural network.

30 Leveraging our surrogate model, we integrate *drag*¹ *guidance* into text-to-image generation models
31 (specifically Stable Diffusion), and show that this integrated system can generate car images that both
32 match a text prompt and minimize predicted drag. We also find that drag guidance can produce more
33 aesthetically streamlined outputs with lower ride heights. Figure 1 contains illustrative examples of
34 our results compared to baseline images generated by an unmodified version of Stable Diffusion.

¹In this paper, we use “drag” and “drag coefficient” interchangeably.

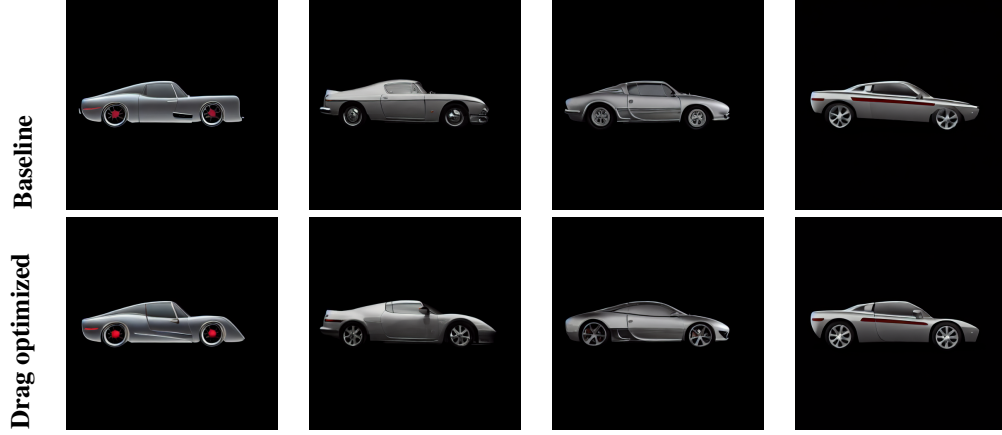


Figure 1: Outputs generated by Stable Diffusion with and without our drag-guidance technique.

2 Method and implementation

Our method consists of (1) a differentiable surrogate model trained to estimate the drag of a generated image of a car, and (2) a modified diffusion sampling process that uses the surrogate model as a guide.

We train a surrogate model that predicts drag coefficients from side-view vehicle images (Figure 2). We use the dataset [31], which consists of 2D vehicle renderings labeled with drag coefficients calculated from CFD simulations. Our model consists of a trainable linear layer attached to a frozen pre-trained feature extractor, a well-established technique in transfer learning for improving robustness to out-of-distribution shifts [16]. Details of the training process can be found in Appendix D.

Note that the surrogate model alone is not sufficient to perform drag optimization. Direct gradient steps over the pixels of an image will cause the image to degrade into noise, likely moving into the adversarial space of the surrogate model (see Figure 3a). We remedy this using a procedure that interleaves drag minimization steps with denoising steps. The details of this procedure are elaborated in Section 2.2. Implementation details can be found in Appendix D, and we evaluate the distributional robustness of our drag estimator in Appendix E.

2.1 Diffusion training and sampling

Diffusion models estimate a *noise vector* $\epsilon \in \mathbb{R}^n$ from a given $y \in \mathbb{R}^n$ and *noise level* $\sigma > 0$ such that $y = x + \sigma\epsilon$ approximately holds for some x in the data manifold. The learned estimator, denoted $\epsilon_\theta : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$, is called a *denoiser*. The trainable parameters, denoted jointly by $\theta \in \mathbb{R}^m$, are found by (approximately) minimizing the loss

$$L(\theta) := \mathbb{E}_{x, \sigma, \epsilon} \|\epsilon_\theta(x + \sigma\epsilon, \sigma) - \epsilon\|^2. \quad (1)$$

Let $\{\sigma_t\}_{t=0}^T$ denote a sequence of noise levels used by *sampling algorithms* to construct novel images using a trained denoiser ϵ_θ . We use the DDIM sampler [32] which, given $x_T \sim \mathcal{N}(0, \sigma_T^2 I)$, generates

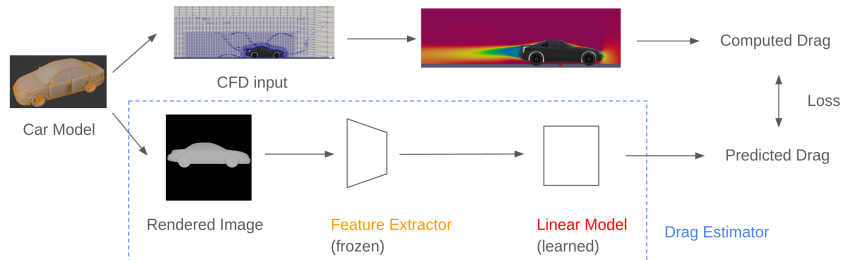
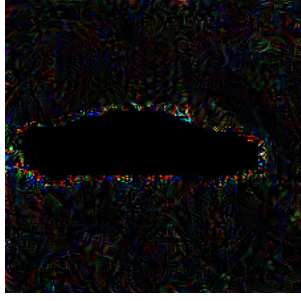


Figure 2: Training the drag estimator

(a) Naive drag optimization on pixel space



(b) Drag optimization interleaved with denoising

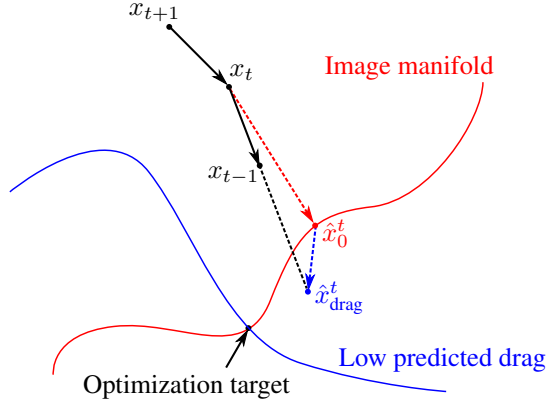


Figure 3: An illustration of how drag guidance works. Noisy images produced by naively minimizing the surrogate model (Figure 3a) are avoided by the inclusion of denoising steps (Figure 3b).

an image x_0 via the following recursion²

$$x_{t-1} = x_t - (\sigma_t - \sigma_{t-1})\epsilon_\theta(x_t, t). \quad (2)$$

2.2 Guiding diffusion

A surrogate model alone is not sufficient to perform drag optimization: starting with an image of a car and minimizing the predicted drag in pixel space produces results similar to Figure 3a.

Let $\phi_{\text{drag}} : \mathbb{R}^{C \times H \times W} \rightarrow \mathbb{R}$ denote our surrogate model. To implement drag-guided diffusion, we follow [8, Algorithm 2] and modify the sampling step (2) to

$$x_{t-1} = x_t - (\sigma_t - \sigma_{t-1})(\epsilon_\theta(x_t, t) + \eta_t \nabla \phi_{\text{drag}}(\hat{x}_0^t)), \quad (3)$$

where η_t denotes an iteration-dependent weight and $\hat{x}_0^t$ denotes the denoised output, defined as $\hat{x}_0^t = x_t - \sigma_t \epsilon_\theta(x_t, t)$. The main difference to the procedure in [8] is that we use a single robust drag estimator that predicts drag for each $\hat{x}_0^t$ instead of training a separate drag estimator for each noise level t . Per [21], we can interpret the construction of $\hat{x}_0^t$ as an approximate projection onto the image manifold. The iterations (3) in turn can be reinterpreted as a damped form of projected gradient descent, an algorithm that alternates between projection and gradient steps. Precisely, (3) implicitly constructs a shadow iterate $\hat{x}_{\text{drag}}^t$ using a projected-gradient descent iteration, and then dampens the update to $\hat{x}_{\text{drag}}^t$ using the quantity $\alpha_t := 1 - \sigma_{t-1}/\sigma_t$.

Proposition 2.1. *The iterations (3) are equivalent to*

$$\begin{aligned} \hat{x}_0^t &= x_t - \sigma_t \epsilon_\theta(x_t, t) \\ \hat{x}_{\text{drag}}^t &= \hat{x}_0^t - \gamma_t \nabla \phi_{\text{drag}}(\hat{x}_0^t) \\ x_{t-1} &= (1 - \alpha_t)x_t + \alpha_t \hat{x}_{\text{drag}}^t \end{aligned} \quad (4)$$

where $\alpha_t = 1 - \frac{\sigma_{t-1}}{\sigma_t}$ and $\gamma_t = \sigma_t \eta_t$.

In total, we can interpret construction of $\hat{x}_0^t$ as stepping towards the image manifold, and construction of $\hat{x}_{\text{drag}}^t$ as a drag-minimization step. This is illustrated by Figure 3b.

3 Experimental results

3.1 Generation of novel designs

We first consider generation of novel vehicle images with drag guidance. Compared to baseline generation, we find that drag guidance can change the generated image in subtle ways (e.g. artifact

²We note [32] presents this recursion in coordinates $z_t := \sqrt{\alpha_t} x_t$ where α_t satisfies $\sigma_t^2 = (1 - \alpha_t)/\alpha_t$.

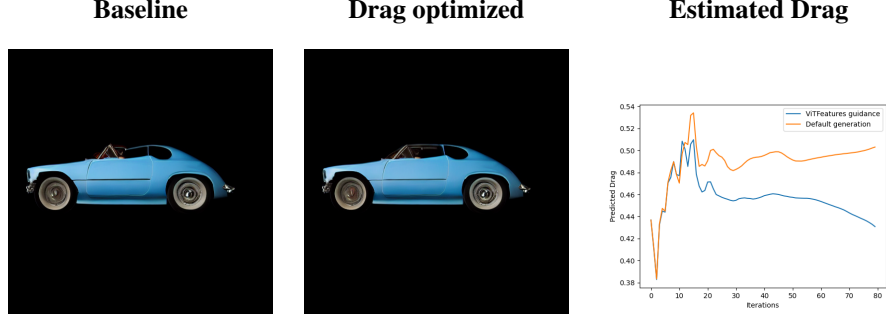


Figure 4: Outputs of our image generation pipeline with drag optimization, compared to baseline images with the drag optimization turned off. Visually, the roof of the generated vehicle is flattened after drag guidance. The plot confirms that the estimated drag is decreased.

removal or surface smoothing as in Figure 4) or through significant redesigns. In all cases, however, a coherent vehicle shape is preserved, as is the artistic style, showing that text-based guidance can work in concordance with drag optimization. Sample results are shown in Figure 4. Figure 4 also confirms that drag guidance indeed decreases predicted drag. Appendix G contains more examples.

3.2 Distributional robustness of drag estimator

Our drag estimator ϕ_{drag} should generalize to the set of intermediate denoised images $\hat{x}_0^t := x_t - \sigma_t \epsilon_\theta(x_t, t)$ generated by the diffusion process. We performed experiments to evaluate the robustness of ϕ_{drag} as a function of the chosen feature extractor (Section 2). Our experiments showed that using pre-trained features from image recognition models makes our drag estimator generalize to intermediate outputs $\hat{x}_0^t$ from the generation process. Experiment details and results are in Appendix E

4 Conclusion, limitations and future work

This work is a step towards generative AI tools that can incorporate guidance from engineering constraints and performance metrics, which could help accelerate the design process. We demonstrate a way that current image generation techniques can be modified to optimize aerodynamic drag. Other metrics that can be inferred from images could be incorporated using appropriate modifications.

A current limitation of our pipeline is its difficulty controlling—or even measuring—the error of the surrogate model on the final drag-optimized output, since the ground-truth drag coefficient is not uniquely determined nor computable from the generated 2D rendering. Future work could alter our approach in a few ways to obtain a unique ground truth. One possibility is to formulate the problem completely in 2D by replacing our surrogate model—which predicts 3D drag from a 2D rendering—with a model that predicts 2D drag from a planar car. Alternatively, one can formulate the problem completely in 3D by replacing Stable Diffusion with a 3D generative model. Another limitation of our approach is its 2D image domain formulation, which may limit its design applications. Replacing images with CAD or 3D models is a natural next step. Combining parametric 3D-models with image diffusion models is the central task of *score distillation sampling* [22]. Adding physics-based guidance to this technique is an interesting topic for future research.

Despite unavailability of ground-truth, there are instances when inaccuracy of the surrogate model is evident, for example in cases in which the surrogate model predicts dramatically different drag coefficients for two cars with similar outlines. Future work could develop active learning techniques that augment the training set by computing ground-truth drag coefficients for generated vehicles, or consider surrogate models with an implicit physical bias that may lead to more accurate predictions.

References

- [1] ABBAS, A., RAFIEE, A., HAASE, M., AND MALCOLM, A. Geometrical deep learning for performance prediction of high-speed craft. *Ocean Engineering* 258 (2022), 111716.
- [2] ALLEN, K. R., LOPEZ-GUEVARA, T., STACHENFELD, K., SANCHEZ-GONZALEZ, A., BATTAGLIA, P., HAMRICK, J., AND PFAFF, T. Physical design using differentiable learned simulators. *arXiv preprint arXiv:2202.00728* (2022).
- [3] BANSAL, A., CHU, H.-M., SCHWARZSCHILD, A., SENGUPTA, S., GOLDBLUM, M., GEIPING, J., AND GOLDSTEIN, T. Universal guidance for diffusion models. *arXiv preprint arXiv:2302.07121* (2023).
- [4] BAQUE, P., REMELLI, E., FLEURET, F., AND FUA, P. Geodesic convolutional shape optimization. In *International Conference on Machine Learning* (2018), PMLR, pp. 472–481.
- [5] CHUNG, H., KIM, J., MCCANN, M. T., KLASKY, M. L., AND YE, J. C. Diffusion posterior sampling for general noisy inverse problems. *arXiv preprint arXiv:2209.14687* (2022).
- [6] CHUNG, H., SIM, B., RYU, D., AND YE, J. C. Improving diffusion models for inverse problems using manifold constraints. *arXiv preprint arXiv:2206.00941* (2022).
- [7] CROWSON, K. CLIP guided diffusion, 2021.
- [8] DHARIWAL, P., AND NICHOL, A. Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems* 34 (2021), 8780–8794.
- [9] DOSOVITSKIY, A., BEYER, L., KOLESNIKOV, A., WEISSENBORN, D., ZHAI, X., UNTERTHINER, T., DEGHANI, M., MINDERER, M., HEIGOLD, G., GELLY, S., ET AL. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).
- [10] DURASOV, N., LUKOYANOV, A., DONIER, J., AND FUA, P. Debosh: Deep bayesian shape optimization. *arXiv preprint arXiv:2109.13337* (2021).
- [11] GIANNONE, G., AND AHMED, F. Diffusing the optimal topology: A generative optimization approach. *arXiv preprint arXiv:2303.09760* (2023).
- [12] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.
- [13] HO, J., AND SALIMANS, T. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022).
- [14] JACOB, S. J., MROSEK, M., OTHMER, C., AND KÖSTLER, H. Deep learning for real-time aerodynamic evaluations of arbitrary vehicle shapes. *arXiv preprint arXiv:2108.05798* (2021).
- [15] KIM, G., KWON, T., AND YE, J. C. DiffusionCLIP: Text-Guided Diffusion Models for Robust Image Manipulation.
- [16] KUMAR, A., RAGHUNATHAN, A., JONES, R., MA, T., AND LIANG, P. Fine-tuning can distort pretrained features and underperform out-of-distribution. *arXiv preprint arXiv:2202.10054* (2022).
- [17] LIU, R., WU, R., VAN HOORICK, B., TOKMAKOV, P., ZAKHAROV, S., AND VONDRICK, C. Zero-1-to-3: Zero-shot one image to 3d object. *arXiv preprint arXiv:2303.11328* (2023).
- [18] MAZÉ, F., AND AHMED, F. Topodiff: A performance and constraint-guided diffusion model for topology optimization. *arXiv preprint arXiv:2208.09591* (2022).
- [19] MENG, C., HE, Y., SONG, Y., SONG, J., WU, J., ZHU, J.-Y., AND ERMON, S. Sdedit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations* (2021).
- [20] NICHOL, A., DHARIWAL, P., RAMESH, A., SHYAM, P., MISHKIN, P., MCGREW, B., SUTSKEVER, I., AND CHEN, M. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models.
- [21] PERMENTER, F., AND YUAN, C. Interpreting and improving diffusion models using the Euclidean distance function. *arXiv preprint arXiv:2306.04848* (2023).
- [22] POOLE, B., JAIN, A., BARRON, J. T., AND MILDENHALL, B. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988* (2022).
- [23] RADFORD, A., KIM, J. W., HALLACY, C., RAMESH, A., GOH, G., AGARWAL, S., SASTRY, G., ASKELL, A., MISHKIN, P., CLARK, J., ET AL. Learning transferable visual models from natural language supervision. In *International conference on machine learning* (2021), PMLR, pp. 8748–8763.
- [24] RAHIMI, A., AND RECHT, B. Random features for large-scale kernel machines. *Advances in neural information processing systems* 20 (2007).
- [25] RAMESH, A., DHARIWAL, P., NICHOL, A., CHU, C., AND CHEN, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125* (2022).

- 163 [26] REMELLI, E., LUKOIANOV, A., RICHTER, S., GUILLARD, B., BAGAUTDINOV, T., BAQUE, P., AND
164 FUA, P. Meshsdf: Differentiable iso-surface extraction. *Advances in Neural Information Processing*
165 *Systems 33* (2020), 22468–22478.
- 166 [27] ROMBACH, R., BLATTMANN, A., LORENZ, D., ESSER, P., AND OMMER, B. High-resolution image
167 synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision*
168 *and Pattern Recognition* (2022), pp. 10684–10695.
- 169 [28] ROSSET, N., CORDONNIER, G., DUVIGNEAU, R., AND BOUSSEAU, A. Interactive design of 2d car
170 profiles with aerodynamic feedback. In *Computer Graphics Forum* (2023), vol. 42.
- 171 [29] SAHA, S., RIOS, T., MINKU, L. L., STEIN, B. V., WOLLSTADT, P., YAO, X., BACK, T., SENDHOFF,
172 B., AND MENZEL, S. Exploiting generative models for performance predictions of 3d car designs. In
173 *2021 IEEE Symposium Series on Computational Intelligence (SSCI)* (2021), IEEE, pp. 1–9.
- 174 [30] SAHARIA, C., CHAN, W., SAXENA, S., LI, L., WHANG, J., DENTON, E., GHASEMIPOUR, S. K. S.,
175 AYAN, B. K., MAHDAVI, S. S., LOPES, R. G., ET AL. Photorealistic text-to-image diffusion models with
176 deep language understanding. *arXiv preprint arXiv:2205.11487* (2022).
- 177 [31] SONG, B., YUAN, C., ARECHIGA, N., PERMENTER, F., AND AHMED, F. Surrogate modeling of car
178 drag coefficient with depth and normal rendering. In *Proceedings of the ASME 2023 IDETC/CIE* (2023).
- 179 [32] SONG, J., MENG, C., AND ERMON, S. Denoising diffusion implicit models. *arXiv preprint*
180 *arXiv:2010.02502* (2020).
- 181 [33] THUEREY, N., WEISSENOW, K., PRANTL, L., AND HU, X. Deep learning methods for reynolds-averaged
182 navier–stokes simulations of airfoil flows. *AIAA Journal* 58, 1 (2020), 25–36.
- 183 [34] UMETANI, N., AND BICKEL, B. Learning three-dimensional flow for interactive aerodynamic design.
184 *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–10.



(a) “a house with two windows”



(b) “a car with a drag coefficient of 0.5”

Figure 5: Illustrates the inability of text-guidance to realize quantitative design specifications. Images were generated by craiyon.com

A Text guidance is insufficient for engineering constraints

Given their capabilities, it is natural to ask how current image generation tools can be applied to engineering design, and, in particular, if text-guidance is sufficient to specify quantitative engineering constraints. Simple experiments reveal significant challenges. For example, given a text prompt of “a house with *two* windows”, a text-guided diffusion model produces a house with windows, but ignores the quantitative constraint on the specified *number* of windows (Figure 5a). The prompt “a car with a drag coefficient of 0.5” is also completely misinterpreted (Figure 5b), showing an inability of the generation process to even correctly parse the constraint.

B Related Work

Guided diffusion Guidance aims to minimize an auxiliary loss function $\phi(x)$ during the image generation process [3]. This is done by incorporating the gradient of $\phi(x)$ into each generation step. Guidance techniques can be differentiated by the type of loss function. In *classifier-based* guidance [8], the loss function $\phi(x)$ is the logit probability of a specified class in a pre-trained classifier. The aim of this guidance is to steer the output towards this class. In *text-based* guidance, $\phi(x)$ is a similarity score computed in a latent space, e.g., the cosine similarity between the CLIP embedding of the image x and a given text-prompt [23, 7, 20, 15]. Guidance can also be used to solve linear equations $Ax = b$ by setting $\phi(x) = \|b - Ax\|^2$. This setup generalizes image inpainting, compressed sensing, and super-resolution [6, 5]. Alternatively, one can implement *classifier-free* guidance, in which gradients of $\phi(x)$ are effectively learned during training. Such approaches have been used for text guidance [13], inpainting [27] and single-view reconstruction [17].

Diffusion for design Prior applications of diffusion to engineering design include [11, 18], which train *new* diffusion models for a specific design task. In contrast, our work focuses on adapting an *existing* diffusion model (Stable Diffusion), aiming to leverage the web-scale dataset used for its training.

Drag optimization and machine learning Drag optimization is a classical problem that can be solved using PDE-optimization techniques. These approaches require accurate solutions of the Navier-Stokes equations, which are computationally expensive to obtain and sensitive to shape representation. Circumventing these difficulties using ML-techniques is an active area of research. One thread of research replaces direct solution of the Navier-Stokes equations with a *surrogate model*. These models are trained using computational fluid-dynamics (CFD) simulations and either predict pressure or velocity fields [34, 26, 2, 10, 4, 1], and/or a performance measure (e.g., drag coefficient) directly from model parameters [31, 4]. Different model architectures have been used, including Gaussian process regression [34], graph neural networks [2, 10], geodesic convolutional neural networks [4, 1], variational autoencoders (VAEs) [29], and UNets [33, 14]. Our work continues this thread and develops a surrogate model for drag coefficient estimation using *linear regression* on features extracted from pretrained neural-networks. Another thread concerns novel shape representations that maintain end-to-end differentiability when cascaded with a surrogate model. Such representations include Deep Implicit Fields [26], PolyCubes [34], and 2D renderings [28, 33, 14, 31]. Our work uses 2D renderings proposed by [31], which capture the point-wise depth and normal information of

three dimensional (3D) shapes using RGB color channels of pixel data. This image representation allows us to easily implement a differentiable surrogate-model that can predict drag coefficients directly from intermediate Stable Diffusion outputs.

C Denoising diffusion models

Diffusion models are a class of generative models inspired by the physical process of diffusion. These models are trained by adding progressively larger amounts of Gaussian noise to a given data set, mimicking a diffusion process. Sampling reverses this process through *denoising*.

C.1 Training

Diffusion models estimate a *noise vector* $\epsilon \in \mathbb{R}^n$ from a given $y \in \mathbb{R}^n$ and *noise level* $\sigma > 0$ such that $y = x + \sigma\epsilon$ approximately holds for some x in the training set. The learned estimator, denoted $\epsilon_\theta : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$, is called a *denoiser*. The trainable parameters, denoted jointly by $\theta \in \mathbb{R}^m$, are found by (approximately) minimizing

$$L(\theta) := \mathbb{E}_{x, \sigma, \epsilon} \|\epsilon_\theta(x + \sigma\epsilon, \sigma) - \epsilon\|^2 \quad (5)$$

when x is drawn from the training-set distribution, σ is drawn uniformly from a finite set of positive numbers, and ϵ is drawn from a Gaussian distribution $\mathcal{N}(0, I)$.

Current state-of-the-art diffusion models modify this training process in a few ways. Models supporting classifier-free text guidance add text embeddings (from CLIP, for example) to the denoiser input [23]. Models supporting high-resolution image synthesis implement *latent diffusion* [27], i.e., the variable x is not an element of pixel space, but, rather, an element of a lower-dimensional latent space defined by a VAE.

C.2 Sampling

Let $\{\sigma_t\}_{t=0}^T$ denote a set of noise levels σ used in training and assume that $\sigma_t > \sigma_{t-1}$. The sequence σ_t is used by *sampling algorithms* to construct novel images using a trained denoiser ϵ_θ . We use the DDIM sampler [32] which, given $x_T \sim \mathcal{N}(0, \sigma_T^2 I)$, generates an image x_0 via the following recursion³

$$x_{t-1} = x_t - (\sigma_t - \sigma_{t-1})\epsilon_\theta(x_t, t). \quad (6)$$

Classifier-free text-guidance [13] executes this recursion using a modified denoiser $\epsilon_\theta(x_t, t, y)$ that takes a reference latent embedding y (e.g., from CLIP) as an optional third input. Concretely, it constructs $\epsilon_\theta(x_t, t)$ via

$$\epsilon_\theta(x_t, t) = (1 - w)\epsilon_\theta(x_t, t, \emptyset) + w(\epsilon_\theta(x_t, t, y)) \quad (7)$$

where w is a weighing factor and $\emptyset$ denotes the empty string.

C.3 Proof of Proposition 2.1

Proof. Expanding and simplifying the last equation gives

$$\begin{aligned} x_{t-1} &= (1 - \alpha_t)x_t + \alpha_t[x_t - \sigma_t\epsilon_\theta(x_t, t) - \gamma_t\nabla\phi_{\text{drag}}(\hat{x}_0^t)] \\ &= x_t + \alpha_t[\sigma_t\epsilon_\theta(x_t, t) - \gamma_t\nabla\phi_{\text{drag}}(\hat{x}_0^t)]. \end{aligned}$$

Equating with (3) shows that $\sigma_t - \sigma_{t-1} = \alpha_t\sigma_t$ and $(\sigma_t - \sigma_{t-1})\eta_t = \alpha_t\gamma_t$. Rearranging proves the claim. $\square$

D Implementation details

To train our model, we used side-view renderings from the dataset [31]. For each item in this dataset, we randomly generated 10 new training examples using horizontal flips, vertical shifts (up to 25

³We note [32] presents this recursion in coordinates $z_t := \sqrt{\alpha_t}x_t$ where α_t satisfies $\sigma_t^2 = (1 - \alpha_t)/\alpha_t$.

Features	Dimension	Train R^2	Test R^2	Test MSE
CLIP	512	0.639	0.545	0.00251
ViT	768	0.639	0.549	0.00249
ResNet	2048	0.630	0.509	0.00271
Random	2560	0.640	0.554	0.00246

Table 1: Performance of drag estimator with different feature extractors.

pixels) and jitter (up to five percent) in brightness, contrast, saturation and hue. The renderings were also resized to 224 by 224 pixels to match the input size of the pre-trained feature extractors, which we obtained from the following Huggingface repositories:

`openai/clip-vit-base-patch32`, `google/vit-base-patch16-224-in21k`, `microsoft/resnet-50`.

We also compared performance to random convolutional features, which we constructed using a kernel size of five and weights drawn from a standard normal distribution. At the output of these convolutions, we applied a bias of +2, a ReLU nonlinearity, and a pooling operation using a 55 by 55 window. After extracting the features, we use ridge regression with regularization parameter $\lambda = 100$ and $\lambda = 10$, for the pre-trained and random features respectively, to train the last linear layer.

Table 1 compares model performance for different feature extractors, specifically, CLIP [23], ResNet [12], Vision Transformers [9], and random convolutions [24].

Finally, drag guidance was incorporated into a checkpoint of Stable Diffusion available at `runwayml/stable-diffusion-v1-5`, with $\eta_t = \eta_0 / \sqrt{1 + 1/\sigma_t^2}$, and $\eta_0 = 400$. During sampling, we fix a reference image x_0 , which is a side-view rendering of a car, and set $T = 80$. Then we let $x_T = x_0 + \sigma_T \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$, a commonly used technique in image-to-image generation [19]. We also fix a text prompt and use a scale of 7.5 for classifier-free text guidance.

E Distributional robustness of drag estimator

Our drag estimator ϕ_{drag} should generalize to the set of intermediate denoised images $\hat{x}_0^t := x_t - \sigma_t \epsilon_\theta(x_t, t)$ generated by the diffusion process. There are two forms of distributional shift that inhibit good generalization. First, the dataset used to train the denoiser ϵ_θ differs from the dataset used to train ϕ_{drag} . Second, the distribution of the denoised outputs $\hat{x}_0^t$ differs from that of the ϵ_θ -training set, given that the denoiser is imperfect, i.e., given that $\epsilon \neq \epsilon_\theta(x + \sigma_t \epsilon, t)$. While it is difficult to evaluate this first type of distributional shift, robustness to the second type can be evaluated with controlled experiments. We carry out two such experiments to evaluate the robustness of ϕ_{drag} as a function of the chosen feature extractor (Section 2). We also note that the first type of shift may be mitigated by retraining ϕ_{drag} using vehicle images from the Stable Diffusion training set, but defer this to future research.

In our first experiment, we take an image used to train ϕ_{drag} and add varying levels of noise. We then predict the drag of the different denoised images $\hat{x}_0^t$. A visualization of $\hat{x}_0^t$ and drag predictions for different noise levels on a single example image is shown in Figure 6. We see that the drag estimator can over or underestimate the predicted drag, and that the error increases with the magnitude of added noise. Next, we perform the same experiment over the entire training dataset, and plot the results in Figure 7. The generalization error is measured with the predicted mean-squared-error of the drag coefficient, which increases with the amount of noise added. We see that the features obtained from pre-trained neural networks generalize much better than the random features, even though they have similar performance on the training data.

F Drag-guided diffusion with a gradient-estimation sampler

A recent variation on DDIM is based on reinterpreting sampling as approximate gradient-descent on the Euclidean distance function of the training set [21]. This leads to a gradient-estimation (GE) sampler that averages $\epsilon_\theta(x_t, t)$ over multiple time-steps, exploiting the fact that the gradient of the

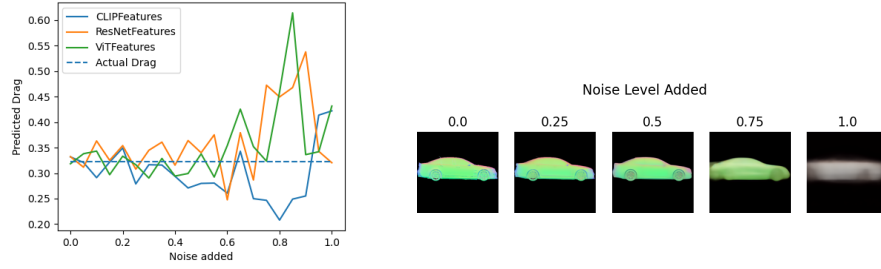


Figure 6: Drag estimator performance on out of distribution data: $\hat{x}_0^t$ with varying amounts of noise added to a single image.

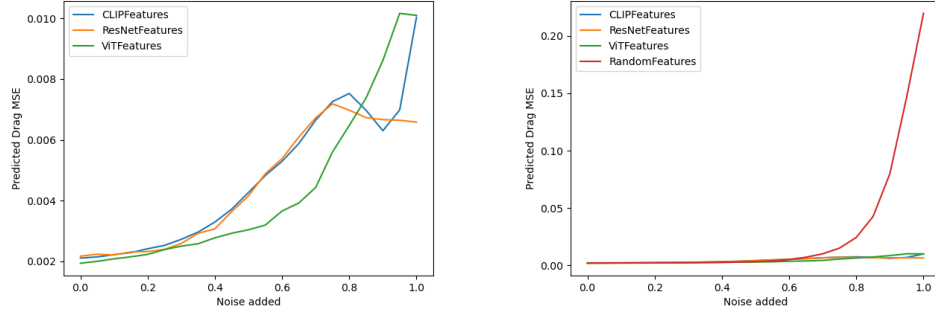


Figure 7: Drag estimator performance on out of distribution data as measured by mean-squared error on the entire dataset. The left plot is a “zoomed-in” version of the right plot, showing that random features lead to a much higher generalization error than features from pre-trained neural networks.

298 distance function is constant along gradient-descent trajectories [21, Section 4]. To use this sampler,
 299 one replaces $\epsilon_\theta(x_t, t)$ in (3) with $\gamma\epsilon_\theta(x_t, t) + (1 - \gamma)\epsilon_\theta(x_{t+1}, t + 1)$ for chosen $\gamma \in \mathbb{R}$. We compare
 300 outputs between GE and DDIM samplers in Figure 8 using $\gamma = 2$.

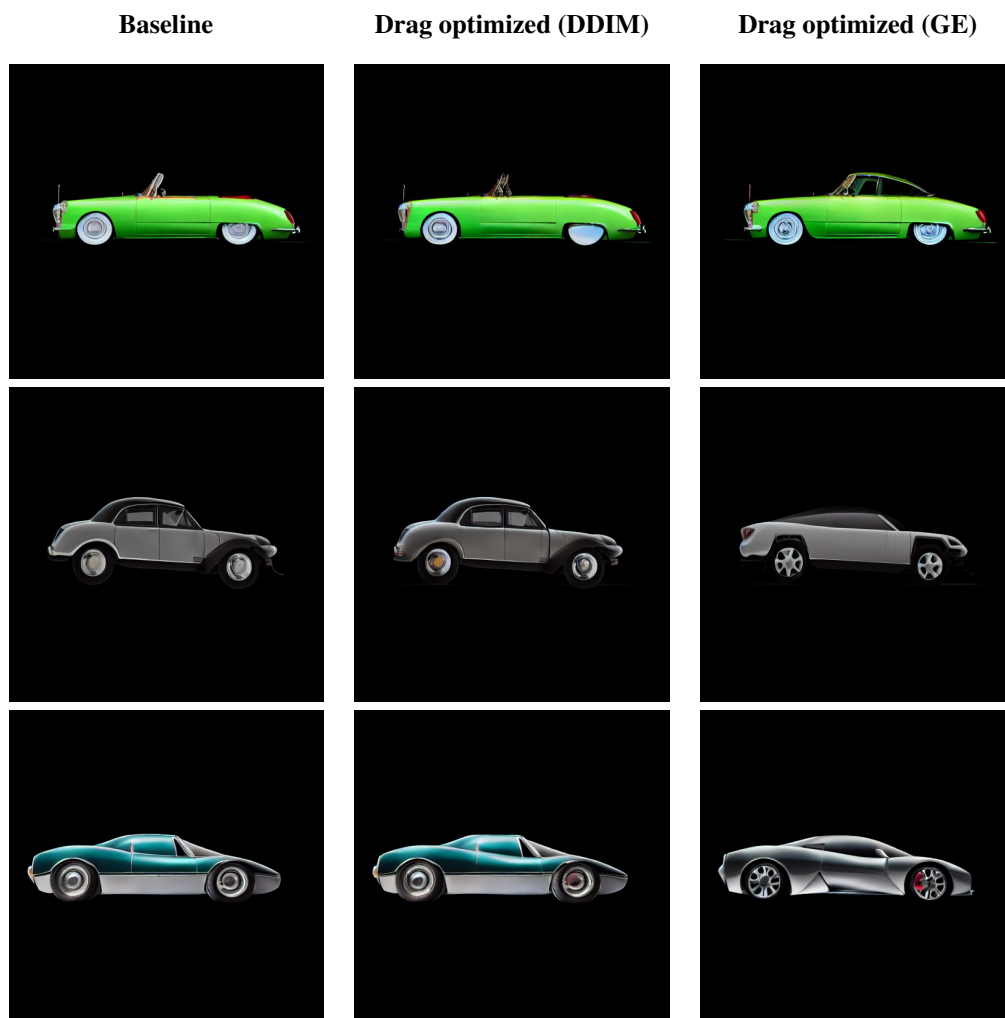


Figure 8: Comparison of DDIM and the gradient-estimation (GE) sampler

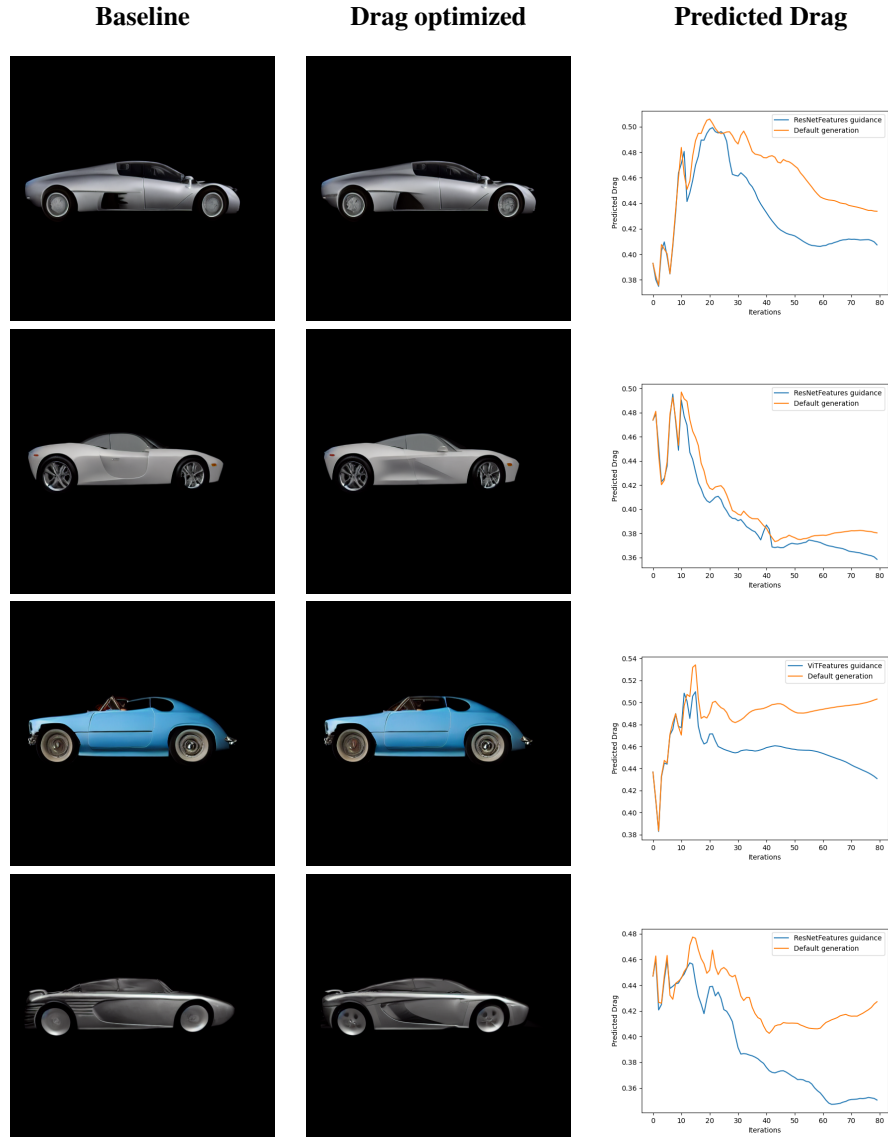


Figure 9: Images generated from the prompt “A car, high resolution studio photo, side view” with and without drag optimization. We also plot drag coefficients predicted by the surrogate model.



Figure 10: Images generated from the prompt “An old fashioned car, high resolution studio photo, side view” with and without drag optimization. We also plot drag coefficients predicted by the surrogate model.



Figure 11: Images generated from the prompt “An iconic car, high resolution studio photo, side view” with and without drag optimization. We also plot drag coefficients predicted by the surrogate model.