Proceedings of the ASME 2024
International Design Engineering Technical Conferences and Computers and Information in Engineering Conference
IDETC/CIE2024
August 25–28, 2024, Washington, DC

DETC2024-XXXX

TO QUANTIZE OR NOT TO QUANTIZE: EFFECTS ON GENERATIVE MODELS FOR 2D HEAT SINK DESIGN

Arthur Drake[1], Jun Wang[2], Qiuyi Chen[1], Ardalan Nejat[3], James Guest[3], Mark Fuge[1,*]

[1]University of Maryland, College Park, MD
[2]Santa Clara University, Santa Clara, CA
[3]Johns Hopkins University, Baltimore, MD

ABSTRACT

In Topology Optimization (TO) and related engineering applications, physics-constrained simulations are often used to optimize candidate designs given some set of boundary conditions. However, such models are computationally expensive and do not guarantee convergence to a desired result, given the frequent non-convexity of the performance objective. Creating data-based approaches to warm-start these models—or even replace them entirely—has thus been a top priority for researchers in this area of engineering design. In this paper, we present a new dataset of two-dimensional heat sink designs optimized via Multiphysics Topology Optimization (MTO). Further, we propose an augmented Vector-Quantized GAN (VQGAN) that allows for effective MTO data compression within a discrete latent space, known as a codebook, while preserving high reconstruction quality. To concretely assess the benefits of the VQGAN quantization process, we conduct a latent analysis of its codebook as compared to the continuous latent space of a deep AutoEncoder (AE). We find that VQGAN can more effectively learn topological connections despite a high rate of data compression. Finally, we leverage the VQGAN codebook to train a small GPT-2 model, generating thermally performant heat sink designs within a fraction of the time taken by conventional optimization approaches. We show the transformer-based approach is more effective than using a Deep Convolutional GAN (DCGAN) due to its elimination of mode collapse issues, as well as better preservation of topological connections in MTO and similar applications.

1. INTRODUCTION

In many engineering problems, high-fidelity methods minimize some performance objective function through a gradient-based iterative process, given an initial set of boundary conditions. Calculating these gradient-based iterations can be computationally expensive. Moreover, human experts are often needed to warm-start the optimization process with initial designs, with the alternative being to use continuation methods that make the initialization choice less impactful—at the cost of significantly more optimization iterations. Thus, the area of Inverse Design (ID), in which problem-specific objectives are directly used to guide the search for optimal designs, has become a rapidly growing subset of machine learning research [1–4]. Specifically, generative models such as Variational Autoencoders (VAEs) [5–9], Generative Adversarial Networks (GANs) [10–14], and Diffusion Models [15–17] are able to quickly produce novel samples after training on a sufficiently large dataset of existing designs. While these models have begun to see success in engineering contexts, the underlying reasons for this success have remained largely unexplored.

One particular area with a large potential to benefit from the resulting balance between physical performance and computational efficiency is Topology Optimization (TO) [18], which aims to optimize material distribution within a given design space to minimize and constrain certain performance objectives of a system. Our paper specifically focuses on two-dimensional heat sink cross-sections with the objective of minimizing mean temperature and fluid power dissipation. While ordinary GANs may seem promising for this work, several potential issues arise upon further inspection. Notably, even a well-performing generator may fail to produce designs without disconnected pockets of fluid—part of a broader issue in which topological features cannot easily be learned by the model. Recent works concerning this issue have examined various loss functions [19] and data representation methods such as connectivity graphs [20] for better topological compliance. Although the results of the latter appear promising, this work remains in its early stages and requires significant data preprocessing for larger datasets.

An important development for generative models has been the adaptation of the popular transformer architecture [21] to model and create complex images whose distributions are otherwise too difficult to learn for typical networks with convolutional architectures. Among the most successful of these

*Corresponding author: fuge@umd.edu

has been the Vector-Quantized GAN (VQGAN), which presents a novel two-stage approach: a reconstructive step that encodes images within a discrete latent space, and a generative step that trains a transformer to learn detailed representations among these discrete latent values [22]. VQGAN has achieved state-of-the-art results in several deep vision benchmarks, but so far has seen limited use in engineering and design applications. In this paper, we show how the quantization mechanism of VQGAN can effectively represent our dataset of two-dimensional heat sinks within a discrete latent space, or *codebook*. We further implement a conditional argument in the transformer stage to autoregressively generate new heat sink designs given a set of three specified boundary conditions.

Overall, the contributions of this paper are as follows:

1. We present a dataset of two-dimensional heat sinks which are generated via a MTO method under various physics-based constraints. Our dataset acts as one of the first benchmarks for modeling binary or grey-scale images using the VQGAN architecture, particularly in the design optimization context.

2. We propose a VQGAN architecture for the inverse design of two-dimensional heat sinks, with a conditional input of desired boundary conditions fed to the transformer stage. We also implement several augmentations to improve training speed and reconstruction accuracy for modeling two-dimensional binary or grey-scale images.

3. We conduct a latent space analysis of several VQGAN variants in comparison to more common generative models, offering insight into how VQGAN can better preserve the topological characteristics of heat sink designs and similar data—even at high compression rates within its codebook.

4. After training on our MTO dataset, we compare the thermal performance of VQGAN-generated heat sinks to conventional generative model outputs, as well as the MTO test data itself. Specifically, we analyze the Pareto fronts of mean temperature and fluid power dissipation for the generated and test data. We further present several topological and statistical metrics to contextualize each model's thermal performance.

2. RELATED WORKS

The following section reviews the related areas of TO dataset generation for heat sinks, augmentation of machine learning frameworks using Vector Quantization, and existing latent space analysis for such frameworks. We supplement this review with background information on the Vector Quantization mechanism itself, and conclude by identifying areas that current research has not yet explored.

2.1 Generating Heat Sink Datasets Using Topology Optimization

Past attempts to create optimal solutions for various heat exchange problems have typically used a parametric optimization scheme alongside a multi-physics approach involving fluid flow and heat transfer approximation [23]. However, these methods are limited in that they usually require a human-provided parameterization of the design space before starting optimization, thus limiting the flexibility of consequent designs. In contrast, Topology Optimization (TO) allows for material to be freely distributed in the design space without a fixed parameterization [24–26]. Koga *et al.* first proposed a complete process for optimizing heat sinks using TO, based on a multi-objective function to minimize pressure drop and maximize heat transfer [27]. Overall, TO has become a popular approach for the multi-physics development of heat sinks [28–31]. These methods have led to improvements in thermal efficiency and other metrics when compared to conventional designs such as straight-fin heat sinks.

While TO methods are generally robust and effective, they are also time-consuming due to the iterative process of approximating solutions. In the case of heat sinks, the nonlinear Navier-Stokes equations and conjugate heat transfer must be addressed [32]. Thus, one may instead desire a machine learning-based approach to generate new heat sinks or similar structures, particularly during earlier design stages. The vast majority of machine learning methods—including the models we discuss in this paper—require a substantial dataset of existing designs in order to produce convincing new outputs. The ability to benchmark the results of different models trained on a diverse standardized dataset is also important. As of the writing of this paper, little research has been conducted to create such a dataset for heat sink designs. Perhaps the closest related work is that of Parrott *et al.* , who utilized existing code to create a dataset of 55,440 coupled thermoelastic structures with randomly assigned loads, volume fractions, and other conditions to train a self-attention-based GAN [33]. In this paper, we create a new, diverse dataset of heat sinks generated via MTO as a novel benchmark for modern generative models, and propose several intuitive topological metrics for evaluation [19]. These metrics may be extended to other applications featuring two-dimensional binary or grey-scale images as well.

2.2 Development of Generative Models Using Vector Quantization

Researchers have increasingly focused on designing generative machine learning architectures to bypass the high computational costs in TO and related applications. Among the most popular of these are Variational Autoencoders (VAEs) [5–9] and Generative Adversarial Networks (GANs) [10–14].

In the context of these generative models, our paper will focus on the data compression method known as Vector Quantization (VQ), which allows for a discrete latent representation of data through optimized embeddings within a container of fixed dimension—known as a codebook [34–36]. The first major implementation of this method in machine learning was the original VQVAE [37]. This model offers several benefits to generative modeling, including the ability to use a more powerful decoder. In ordinary VAEs, doing so would lead to posterior collapse, *i.e.*, the decoder begins to ignore the latents and learns an alternate representation. A subsequent work proposed VQVAE-2 [38], which improved on the previous model by learning local and global information separately through a hierarchical architecture. However, these early models lack a powerful method for autore-

gressive prior modeling, and typically rely on a convolutional method such as PixelCNN [39] to learn and sample from their latent distributions. Thus, Esser *et al.* developed VQGAN, the basis for this paper, which implements a transformer architecture in the second stage to learn the discrete latent codes [22]. The use of a transformer eliminates the inductive bias present in convolutional models, which favor local patterns over more complex relationships in the data. This aspect is particularly important for engineering designs such as heat sinks—which require carefully interconnected components to function optimally. The VQGAN also features augmentations to the loss function and the use of an adversarial training method to promote better-learned representations in the initial convolutional stage. While some incremental improvements have been made to the VQGAN over the last few years [40–42], they offer only marginal benefits considering such models have rarely been applied to the engineering and design fields [43]. In this work, we present several augmentations to the baseline VQGAN architecture to benefit applications featuring binary and grey-scale images, such as our MTO dataset.

2.3 Latent Space Analysis of VQ-Based Generative Models

Despite the recent success of VQGAN and related developments, little work has been conducted to explore the various properties of the resulting discrete latent space. For example, Parrott *et al.* recently proposed several effective GANs for multidisciplinary design optimization (MDO) problems [33, 44], but did not provide a comprehensive latent analysis for their models. Meanwhile, Hu *et al.* inspected the latent space of GANs by proposing a hypothetical distance between the data and latent distributions—whose minimization corresponds to the least complex generator necessary to accomplish this mapping [45]. However, little attention has been given to more practical aspects of the latent space which may be necessary in engineering and design applications. For example, it is interesting to consider whether a continuous manifold may be approximated from a discrete latent space, allowing for the discovery of other near-optimal designs existing on the manifold. In effect, this would mimic the sampling methods in traditional GANs without ever directly learning a continuous latent distribution. Another property to consider is the robustness of the latent space to distortions—for example, the switching of two discrete codes—as measured by their observed impacts when decoded to the data space. In short, drawing connections between the latent mapping and its output conditions is a vital task (albeit an application-specific one) if VQGANs and other generative models are to be adopted for real design generation tasks. In this paper, we uncover the VQGAN's unique latent space properties in the context of two-dimensional grey-scale datasets via training and evaluation on our MTO data.

3. METHODOLOGY

Our paper's methodology involves four steps: (1) generating a diverse set of two-dimensional heat sink designs, (2) training various augmented VQGAN models and associated transformers, (3) investigating properties of each learned latent space, and (4) conducting a statistical and thermal performance analysis for the transformer-generated heat sinks. We group the latter two steps into a separate Experiments section to present a more concise and understandable set of results.

3.1 Dataset Creation via Multiphysics Topology Optimization

The generative models' real dataset consists of optimized topologies of two-dimensional (2D) heat sinks. To create the dataset, we perform MTO over a range of input boundary conditions to achieve high-performing 2D heat sink designs. We adapt an OpenFOAM[1]-based MTO solver developed by Yu *et al.* [46] to perform density-based topology optimization on fluid-thermal problems, which are formulated as follows:

$$\begin{aligned} \min_{\gamma} \quad & \Psi = \frac{1}{|\Omega|}\int_{\Omega} T \mathrm{d}\Omega \\ \text{s.t.} \quad & \frac{1}{|\Omega|}\int_{\Omega} \gamma \mathrm{d}\Omega \leq V, \quad 0 \leq \gamma \leq 1 \\ & J < \bar{J} \\ & Equations\ (6), (7) \end{aligned} \tag{1}$$

where Ψ is the mean temperature, T is the temperature field, and $|\Omega|$ is the volume of the computational domain. The first inequality constraint controls the overall volume fraction V occupied by the fluid. The density field $\gamma \in [0, 1]$ is used to describe the distribution of material, where $\gamma = 0$ represents solid and $\gamma = 1$ represents fluid. The second inequality constraint requires the fluid power dissipation J of the fluid device to be below a certain bound (*i.e.*, $\bar{J}$) to prevent unrealistic designs. We note that in addition to this constraint, we also minimize J itself as part of the multi-objective problem examined in this work.

Besides these inequality constraints, the MTO problem is also subject to two sets of governing equality constraints: the Navier-Stokes equations (Equation (6)) and energy conservation equations (Equation (7)). The equations of the power dissipation J and these equality constraints are detailed in Appendix A.

In this work, we optimize the 2D MTO problem as in Figure 1. The boundary conditions are as follows: (1) Constant velocity and temperature are applied at the inlet, with zero pressure and thermal isolation at the outlet. (2) No-slip adiabatic boundary conditions are enforced on all other external walls. (3) A uniform heat source Q is used throughout the entire domain.

To generate a diverse dataset, we perform MTO over a range of three boundary conditions: Reynolds number (Re), power dissipation ($\bar{J}$), and fluid volume fraction (V). Among the three boundary conditions, Re is indirectly varied using v, which is the inlet velocity[2]. Figure 2 illustrates how the three boundary conditions independently affect the final designs in the 2D MTO problem. To create a larger dataset, we expand the ranges of the

[1] https://www.openfoam.com/. OpenFOAM is the free, open-source CFD software developed to solve anything from complex fluid flows involving chemical reactions, turbulence and heat transfer, to acoustics, solid mechanics and electromagnetics.

[2] $Re = \frac{\rho v L}{\mu}$, where ρ is the water density (1000 kg/m^3), v is the inlet velocity, L is the length of the inlet tube (2×10^{-3} m), and μ is the dynamic viscosity of water (1×10^{-3} $N \cdot s/m^2$). The range of Re is [50, 190]. Therefore, the inlet velocity v has a range [-0.095, -0.025]. The negative sign indicates the velocity direction.

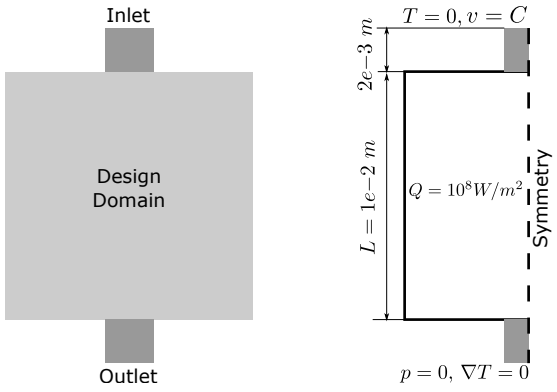

FIGURE 1: THE MTO PROBLEM SETUP FOR 2D HEAT SINK DESIGNS.

three boundary conditions to $J \in [5\underline{J_1}, 75\underline{J_1}]$, $V \in [0.25, 0.7]$, and $Re \in [50, 190]$, where $\underline{J_1} = 1.58 \times 10^{-7}$ is used as a reference value for the power dissipation. We leverage a High-Performance Computing (HPC) cluster to generate the dataset by performing the 2D MTO problem in parallel over 5,000 different boundary condition groupings produced via Latin Hypercube sampling [47]. In the end, we successfully completed 4,579 optimizations, yielding 3,434 training and 1,145 test samples (75/25%). The remaining 421 samples failed to reach the required 200 optimization steps, likely due to numerical instabilities in the solver, and were discarded despite satisfying constraints. We also found about two-thirds of the optimized samples did not fully satisfy the specified power dissipation constraint, although this does not significantly impact our final results. Future work will fix these issues via additional tuning of the MTO solver.

3.2 VQGAN

We now discuss our implementation of the VQGAN, which was originally proposed by Esser *et al.* [22]. We first provide an overview of the architecture and explain its effectiveness in many applications, including inverse design for MTO. We then describe a few limitations of the original implementation and the augmentations we propose to address them. Broadly speaking, VQGAN consists of two main stages: (1) an AutoEncoder (AE)-based first stage whose purpose is to learn an information-rich discrete latent space, or *codebook*, and (2) a second stage in which a transformer is trained to autoregressively predict new images by representing codebook indices as tokens, which may then be decoded back to the original data space.

3.2.1 Model Architecture and Setup. The first stage of VQGAN fundamentally relies on a convolutional encoder E and decoder G, which translate images into the latent space and back. The architectures of these modules are very similar to those of other modern autoencoder-based models, using blocks composed of simple convolutional layers, an activation function such as ReLU, and some form of mini-batch normalization such as GroupNorm [48]. The first novelty of the first stage comes in the discrete codebook Z, which in our case represents an image $x \in \mathbb{R}^{H \times W \times 1}$ as a collection of entries $z_q \in \mathbb{R}^{h \times w \times n_z}$. We note that as opposed to the 3 RGB channels of ordinary color images, our heat sinks only have a single channel representing a range from solid (0) to fluid (1) material. Aside from this, the total number of discrete codes $|Z| = 1024$ is used as a starting point from the original VQGAN implementation. We can thus view the total codebook size as $|Z| * n_z$, given that n_z is the dimension of each code. After the initial image encoding step, a subsequent quantization step $q(\cdot)$ is performed to match each continuous latent code to its Euclidean nearest neighbor in the codebook. The final reconstructed image can then be denoted as $\hat{x} = G(q(E(x)))$. To address the non-differentiable quantization step during backpropagation, the decoder gradients are simply copied to the encoder.

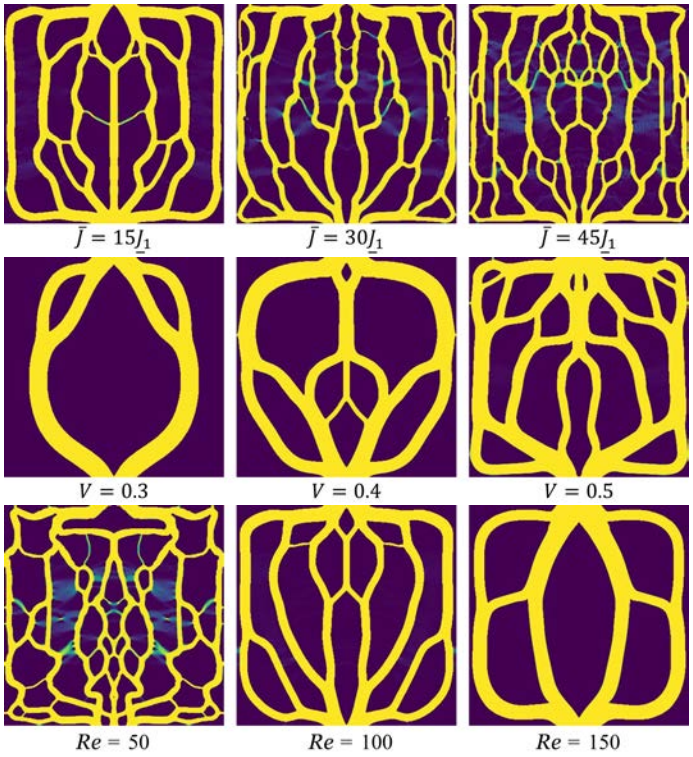

FIGURE 2: TOP ROW: OPTIMAL DESIGNS FOR $\bar{J} = 15\underline{J_1}$, $\bar{J} = 30\underline{J_1}$, AND $\bar{J} = 45\underline{J_1}$ ($Re = 100$ AND $V = 0.4$). MIDDLE ROW: OPTIMAL DESIGNS FOR $V = 0.3$, $V = 0.4$, AND $V = 0.5$ ($\bar{J} = 5\underline{J_1}$ AND $Re = 100$). BOTTOM ROW: OPTIMAL DESIGNS FOR $Re = 50$, $Re = 100$, AND $Re = 150$ ($\bar{J} = 10\underline{J_1}$ AND $V = 0.4$).

During the training phase of the first stage, several considerations are taken to learn a powerful encoder, codebook, and decoder. Firstly, the vector quantization loss originally proposed in VQVAE [37] is as follows:

$$L_{\text{VQ}}(E, G, Z) = L_{\text{rec}} + \|\text{sg}[E(x)] - z_q\|_2^2 + \|\text{sg}[z_q] - E(x)\|_2^2 \quad (2)$$

The first loss term is a reconstruction loss which E and G both work to optimize. In VQGAN, this is composed of a simple L1 loss, as well as a perceptual loss term to increase the perceptual quality of reconstructions [49, 50]. The second loss term freezes the encoded vectors and pushes them toward the codebook vectors, with the opposite occurring in the third term.

In short, E optimizes terms 1 and 3, G optimizes term 1 only, and the codebook Z optimizes term 2.

Furthermore, VQGAN introduces an adversarial training approach using a PatchGAN discriminator [13], allowing for a higher codebook compression rate and therefore shortening the final latent sequence, which is used to train the transformer in Stage 2. The overall objective for Stage 1 is therefore:

$$Q^* = \arg\min_{E,G,Z} \max_{D} \mathbb{E}_{x\sim p(x)} [L_{VQ}(E,G,Z) + \lambda L_{GAN}(\{E,G,Z\},D)] \tag{3}$$

where $L_{GAN}(\{E,G,Z\},D) = \log(D(x)) + \log(1 - D(\hat{x}))$ and λ is an adaptive weight to balance the two main loss terms, as described in [22].

Once the first stage of VQGAN has been trained to produce a well-converged encoder, codebook, and decoder, the transformer stage may then be implemented. In this phase, encoded images within the codebook are represented by their equivalent codebook indices, which indicate the positions of codebook vectors most closely matching the encoded vectors. These indices are then unfolded from their original matrix form to a linear sequence and fed into a transformer for autoregressive modeling. That is, given a sequence of previous tokens, the transformer is tasked with inferring the correct next token. Once a full sequence is predicted, the resulting discrete image z_q may be decoded back to the original data space via $\hat{x} = G(z_q)$ by using the pre-trained decoder from Stage 1. For this paper, we also prepend a conditional term r to the start-of-sequence token, allowing heat sinks to be generated based on the three boundary conditions described in Section 3.1. Since r also needs to be a set of discrete indices, we train another small VQGAN Stage 1 to create a codebook representing the boundary conditions, which we refer to as the Conditional VQGAN (C-VQGAN). We further describe the C-VQGAN implementation in Section 4.

While the default implementation of VQGAN described above is effective for most image reconstruction and generation applications, we provide several minor improvements to the architecture in the context of modeling binary or grey-scale images. This is in contrast to nearly all existing VQGAN applications, which focus on three-channel RGB images. Research areas which could benefit from these changes include but are not limited to: topology optimization (*i.e.*, our MTO dataset), road network modeling, and microstructure generation. In short, we aim to minimize model complexity while preserving or improving reconstruction accuracy. This is motivated by the benefits of reduced training time and computational load, as well as maximized data compression in the codebook (to provide a simpler distribution for the transformer to learn in downstream tasks). Our main changes are as follows:

1. Hidden Layer Modifications (HLM): We reduce the width of hidden layers to a maximum of 256 as compared to 512 in the existing PyTorch implementation[3]. We also remove one of two attention layers (called Non-Local Blocks), keeping only the one between the innermost Residual Blocks of the encoder and decoder. We find that the second attention layer is somewhat redundant and unnecessarily increases model size.

2. No Discriminator (ND): Although a discriminator was proposed in VQGAN as an improvement over VQVAE in terms of perceptual accuracy, we find this addition actually *limits* the expressivity of the network in our case. Notably because our application only features single-channel images and discourages some fine details like disconnected fluid channels or intermediate material, it is actually better to remove adversarial training to stabilize training under the smaller codebooks and encoder/decoder networks we aim to utilize. With this change in mind, we still keep a single-channel perceptual loss (LPIPS) during training to assist in recognizing local connectivity patterns, but remove it as a metric because the underlying VGG network is not trained to properly evaluate such images. As we describe in later sections, a combination of a simple L1 reconstruction loss and more advanced topological metrics are used instead.

3. Increased learning rate (η): Given we train all models for 100 epochs for the sake of comparison, a higher learning rate should intuitively lead to quicker convergence. Unfortunately, we find this is not the case while using the discriminator, as it only destabilizes training further. However, the more stable training regime of ND does benefit from such a change, with η as high as 2e−4 (compared to the original 2.25e−5) leading to faster convergence. Due to the steady increase in codebook utilization over the course of training, this results in a more complete codebook given only 100 epochs.

4. Minimization of n_z and $|Z|$: Following the other modifications, we find the base value of $n_z = 256$ is overly large for the MTO dataset. After halving n_z repeatedly, we observe that a value as low as $n_z = 4$ is acceptable for reconstructing the MTO samples when removing the discriminator. We thus use $n_z = 4$ for all further VQGANs in this paper, noting this value should be carefully chosen based on the dataset used and desired level of reconstruction accuracy. One interesting observation is that lowering n_z also leads to naturally higher codebook usage, due to the constraint on how expressive each individual code can be. We take advantage of this and a higher learning rate to force full codebook utilization at $|Z| = 64$, which is greatly reduced from the unnecessarily high initial value of $|Z| = 1024$. Overall, this results in a dense yet compact latent representation that be learned relatively quickly by a transformer, given its limited dimensionality.

5. Combined model: Considering the above changes, we finally aim to train a single model which benefits from all of them at once. After consideration of a few different combinations, we found the top performer to include HLM, ND, $\eta = 2e{-}4$, $n_z = 4$, and $|Z| = 64$. While valid concerns exist about over-compression and over-simplification of such a combined model, the individual augmentations build on each other rather than interfering in this case. The end result is a tuned

[3] https://github.com/dome272/VQGAN-pytorch

VQGAN that not only reduces L1 error by 20 percent and achieves full codebook utilization, but also trains 40 percent faster than its predecessor, taking under 90 minutes on a single NVIDIA H100 GPU. The general ablation procedure used in this experiment may thus be used as a starting point for future works using VQGAN to model single-channel, topologically connected data.

With these baseline changes implemented, we are now ready to examine the VQGAN latent space properties and transformer generation capability through a series of detailed experiments.

TABLE 1: ABLATION STUDY FOR THE STAGE 1 VQGAN AUGMENTATIONS IN TERMS OF MEAN ABSOLUTE ERROR (MAE), CODEBOOK USAGE, AND TRAINING TIME.

Model	MAE ↓	Code Use (%) ↑	Train (min) ↓
Original	0.105	29 (2.83%)	184
HLM	0.103	33 (3.22%)	**108**
$\lambda_D = 0.1$	0.108	37 (3.61%)	184
$\eta = 2e{-}4$	0.164	128 (12.5%)	184
$\eta = 2e{-}4$, $\lambda_D = 0.1$	0.076	147 (14.4%)	184
$n_z = 16$	0.105	50 (4.88%)	184
$n_z = 16$, $\lambda_D = 0.1$	0.111	49 (4.79%)	184
$\|Z\| = 128$	0.097	40 (31.3%)	184
Combined	**0.074**	**128 (100%)**	**108**

4. EXPERIMENTS

The following section illustrates the effects of including VQ in a generative model through comparisons with traditional AE and GAN architectures that possess only continuous latent spaces. We begin by detailing our setup for the two stages of VQGAN and the continuous models for comparison. For Stage 1, we analyze each model's reconstruction and topological accuracy when compared to the test set. We then present the results of the latent analysis, including interpolation and pixel swapping. For Stage 2, we similarly present the accuracy metrics of each model, supplemented by the results of a warm-start optimization conducted by our MTO solver. This process uses the same termination criteria as the cold-start optimization that created the MTO dataset. Finally, we conduct a Pareto analysis to illustrate the thermal performance of the final generated designs after warm-starting.

4.1 VQGAN Stage 1: Sample Reconstruction And Topological Characteristics

For the reconstructive portion of VQGAN (Stage 1), we begin with the baseline model as proposed in [22] with our three main improvements to accommodate the MTO dataset as described above. We train all models on a NVIDIA RTX 3090 Ti GPU for 50 epochs using the Adam optimizer [51], with batch size 16 and learning rate 1e−3. In our initial training, we observed that each model's loss stabilized at around 20 epochs, with minor improvements thereafter, and thus 50 total epochs were deemed sufficient. In addition, we train a Conditional VQGAN (C-VQGAN) to be incorporated in the transformer (Stage 2). The C-VQGAN simply converts the boundary conditions to a miniature discrete codebook, and is trained in a similar way as VQGAN Stage 1. Precisely, we use 128 embeddings with 4 dimensions each—compared to respective values of 1024 and 16 for the baseline VQGAN. We train this network for 500 epochs, which allows for convergence to a very high reconstruction accuracy given the resulting 4×4 latent feature map. In downstream applications with the VQGAN transformer, we prepend the tokens r generated by C-VQGAN to the start-of-sequence token to enable conditional predictions.

4.1.1 Further Augmentations. The second major set of augmentations to VQGAN involves altering the codebook distribution, followed by an assessment of the latent space quality (Stage 1) and transformer training characteristics (Stage 2). Firstly, to reduce codebook complexity, we implement a two-step training process during Stage 1, which was recently proposed as a "Decoupled Autoencoder" (DAE) method by Hu *et al.* [45]. In the resulting DAE-VQGAN, the encoder and codebook are only trained initially with a weak decoder. After this, a more powerful decoder is trained while the encoder and codebook are held frozen. In this context, we refer to the weak decoder as one that utilizes two-dimensional Dropout [52] during the first half of training epochs. However, other approaches to "weaken" the decoder may be taken, such as halving the number of channels in the hidden layers. The idea motivating this change is that before targeting an accurate reconstruction of the inputs, a simplified yet accurate latent distribution should be prioritized for the encoder. This is because, in a single-stage training process, there exists a trade-off between reconstruction quality (stronger decoder) and latent space efficiency (weaker decoder) [45]. The proposed two-step method therefore allows us to mitigate the effects of this trade-off, resulting in a VQGAN with both desired qualities.

In contrast, one may desire to increase the codebook complexity to model a more difficult dataset and better capture its underlying properties. This may be important for cases like the MTO dataset, in which the presence of even a thin fluid channel might significantly impact thermal performance. We thus utilize a recently-developed improvement for VQ called the "Online Clustered Codebook" [53]. This approach mitigates the common issue of dead codebook vectors by re-initializing these vectors based on anchors sampled from the encoded features. In short, the resulting Online-VQGAN encourages all vectors to move closer to the data distribution and results in full usage of the codebook. We can leverage this benefit further by halving the codebook size relative to the baseline model—which is known to have less than 50% codebook utilization on popular datasets such as ImageNet [53]. In theory, the resulting model should thus still produce good reconstructions despite a lower specified $|Z|$ value. Later analysis will explore the precise meanings of a "good" latent space and "good" reconstruction through comparisons between the baseline VQGAN and the two major augmentations presented here.

For DAE-VQGAN, we utilize the same general structure as the baseline, but impose dropout with $p = 0.4$ in the first 25 epochs. This value of p was found to be more stable for the MTO case than the original $p = 0.5$. We then freeze the encoder and codebook for the final 25 epochs, and allow the decoder to be trained without dropout. For the Online-VQGAN,

we simply replace the baseline codebook with the one developed in [53]. During training, we observe that both the Baseline and the Online models converge fairly quickly, with the latter having a slight advantage in the early stages. Meanwhile, the DAE model experiences some instabilities before settling to an intermediate loss value until 25 epochs, at which point dropout is removed from the decoder and the other components are frozen. The DAE then approaches a loss value very close to that of its counterparts.

In terms of reconstruction performance, there is a significant difference between each of the VQGANs and the AE, as shown in Table 2. None of the VQGANs are able to reproduce the AE's level of perceptual (LPIPS) accuracy. However, their topological accuracy does not suffer as much in comparison, as measured by the mean Solid Segment Error (SSE)[4]. This is particularly true for the Online model, which notably exhibits 100% codebook utilization. Intuitively, the accuracy boost makes sense because the resulting codebook is able to utilize all $|Z| = 1024$ discrete codes, allowing for a more fine-grained latent space with a better ability to reconstruct the original image. Nonetheless, we find that all VQGANs produce satisfactory outputs, both qualitatively and with respect to their measured topological error. Even for the DAE model, most of the error manifests in the form of small fluid channels only.

The true objective then lies in minimizing codebook size while preserving adequate reconstruction quality—whose requirements may vary greatly between different design applications. To better compare the true quality of each latent space, we thus re-train the Online-VQGAN to match the codebook sizes of the Baseline and DAE models. We take this approach because the Online-VQGAN is the only model here with a fully-controllable codebook size. For the Online model trained to be the same size as the Baseline ($|Z| = 359$), we observe the exact same results in terms of both LPIPS and topological error. While these two metrics alone are not enough to fully describe the latent quality, they are a strong indication that both models perform very similarly under such a network size constraint. On the other hand, the Online-VQGAN trained to mimic the DAE-VQGAN ($|Z| = 178$) performs slightly better in the perceptual loss category, yet fares worse in topological accuracy. It is therefore possible that the DAE model learns a more complex topological representation in its first 25 epochs of training, when dropout is imposed on the decoder and prior to the encoder and codebook being frozen. We further examine this hypothesis in our latent space testing in Section 4.2. From now on, we refer to the first reduced-size Online-VQ model as "Online-359" and the second as "Online-178," from their respective number of embeddings. We will also observe the downstream impacts of codebook size and quality in Section 4.3, in which we train a transformer on the VQGAN latent space as described previously.

4.2 Latent Space Analysis

To understand how VQ impacts the learned reconstructive model, we explore the latent space of each VQGAN, comparing its properties to those of a continuous Deep AutoEncoder (AE). We evaluate the latent properties in terms of continuity and robustness to random changes, using both pixel-based and topological metrics. Our results provide a convincing case for the VQGAN architecture despite its minor shortcomings in pure reconstruction ability. To make fair comparisons, we implement the same encoder and decoder architecture in the AE as in VQGAN Stage 1, with the same loss functions used during training. This essentially renders the AE an equivalent model (minus the discrete codebook).

[4]For an explanation of SSE and other topological metrics we use in this paper, please refer to Appendix B.

TABLE 2: RECONSTRUCTION METRICS AND CODEBOOK USAGE RATES FOR EACH VQGAN TESTED.
***MODEL WITH SAME CODEBOOK SIZE AS BASELINE VQGAN.**
****MODEL WITH SAME CODEBOOK SIZE AS DAE-VQGAN.**

Model	LPIPS ↓	SSE (%) ↓	Codebook Use (%) ↑
AE	0.034	3.2	n/a
Baseline	0.128	6.3	(359/1024) 35.1
DAE-VQ	0.158	7.3	(178/1024) 17.4
Online-VQ	**0.117**	**5.4**	**(1024/1024) 100.0**
Online-VQ*	0.128	6.3	**(359/359) 100.0**
Online-VQ**	0.142	8.7	**(178/178) 100.0**

4.2.1 Interpolation.

In many engineering contexts, interpolation between designs yielding other valid designs is critical to generate a diverse range of initial candidates—which may have otherwise not been found in traditional optimization approaches. For example, the optimal heat sink may be difficult to predict via expert analysis or conventional TO. In this paper, we desire two main properties of interpolated samples:

1. **Topological continuity:** Fluid channels should remain interconnected for as long as possible, without any completely disconnected fluid segments. The number of solid segments should steadily change as the fluid surrounding them shifts.
2. **Lack of intermediate values:** All pixels should remain close to 0 or 1, indicating topological certainty, unless an end design contains substantial amounts of intermediates.

Beginning with topological continuity, we emphasize the minimization of disconnected fluid segments. We consider two initial, model-generated reconstructions that only feature a single interconnected fluid channel. While interpolation in the AE latent space is trivial, the process for the VQGAN is restricted by its discrete codebook. We thus interpolate values directly in the VQGAN encoded space before quantizing and decoding them back to the data space.

To better represent how fluid material is distributed, we round up any pixel value of 0.2 or higher to a value of 1 (fluid), while all other values are rounded to 0 (solid). The value of 0.2 is chosen heuristically by observing the AE's interpolation behavior. Our objective in this experiment is to produce a curve which is as close to a value of 1 as possible for the entire interpolation. In effect, the integral of this curve can be seen as a scalar performance metric. Figure 3 shows the results for the AE and each VQGAN. One can observe that the AE reaches by far the most disconnected channels

on average, with a sharp but topologically unstable shift in the design. Though not possible to display within this manuscript, we confirm the phenomenon by observing an animation of the design changing in real time, which shows that a large number of intermediate values appear between the 40th and 60th steps. Because these intermediates are not well-connected, they produce many wasteful fluid pockets when their pixel values are rounded up.

In contrast, all VQGANs fall well below the AE curve, with the Online model generally exhibiting the best topological behavior. Interestingly, the Online model goes against our previous expectations of worsening topological performance with decreasing codebook size; the Online-178 variant actually yields the *fewest* disconnected channels in the middle stages of interpolation. This is in stark contrast to the DAE-VQGAN, which has the same latent dimensions and yet produces double the number of disconnects on average. We can now see that pure topological reconstruction accuracy—as presented in Table 2—does not necessarily relate to the robustness of the learned topological representation. This mismatch may be occurring because the topological accuracy metric is heavily affected by thin channels, which generally have little effect on thermal performance anyway. We thus find the interpolation test to be superior in determining the true level of fluid channel connectivity learned by the VQGAN.

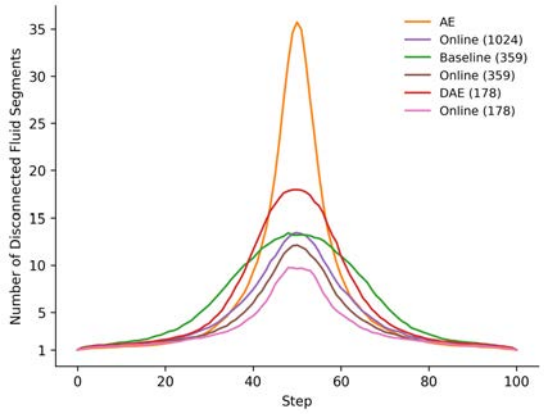

FIGURE 3: AVERAGE NUMBER OF DISCONNECTED FLUID CHANNELS DURING INTERPOLATION WITH CODEBOOK SIZE REPRESENTED BY $|\mathcal{Z}|$ IN PARENTHESES.

In the second experiment, we directly measure the amount of intermediate values produced during interpolation. We measure the proportion of pixels in the design with values between 0.2 and 0.8—representing values with the highest uncertainty in the final (binary) design. As shown in Figure 4, the AE once again reaches the highest value of any model, followed by the DAE and Baseline VQGAN. Meanwhile, the Online models feature the lowest variation in this value between the ends and middle of the interpolation. However, the Online-178 model suffers a one-percent increase in its initial amount of intermediate values compared to its larger counterparts, suggesting its pure reconstruction ability has started to fall behind given its small codebook.

Finally, we examine the average pixel-wise rate of change across the heat sink during interpolation. A higher rate of change

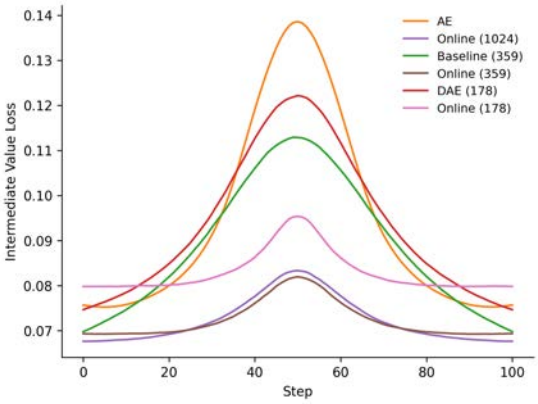

FIGURE 4: AVERAGE INTERMEDIATE VALUE PENALTY INCURRED DURING INTERPOLATION.

near the start or end of interpolation suggests instability, particularly in the VQGAN case. This is because small changes in the continuous encoded space do not necessarily represent small changes within the codebook (and thus the decoded design as well). The issue results in fluid channel structures fluctuating at very early or late interpolation steps. While this likely does not result in much thermal instability, it is still an undesirable behavior. Figure 5 shows that the smaller codebooks of the Baseline and DAE-VQGAN exacerbate this problem. However, the Online models suffer less as their codebook size decreases, particularly the Online-359 variant. This model provides the best stability at either end of the interpolation—approaching that of the AE—with a sharper upward curve in the center indicating a more defined topological change.

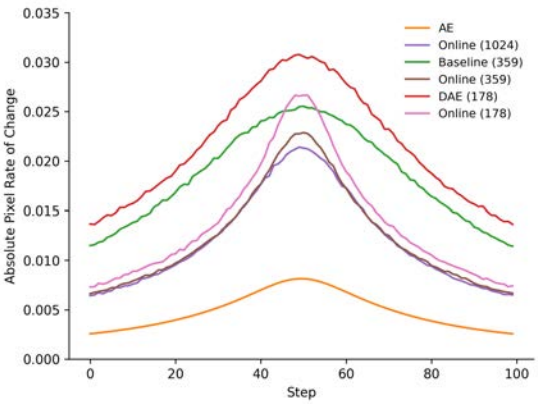

FIGURE 5: AVERAGE ABSOLUTE RATE OF PIXEL CHANGE DURING INTERPOLATION.

Overall, the above interpolation results are an encouraging sign for the topological capabilities of the VQGAN—particularly for the Online Codebook augmentation. Given an equal latent size to the AE and a sufficiently large hidden dimension, the VQGAN is able to provide significant topological (and as a result, thermal) benefits when exploring the latent space for new heat sink designs. The decrease in intermediate material similarly allows for more

clear-cut designs and reduces the need for post-processing. We can infer that despite the two smaller Online-VQGANs having equivalent codebook sizes to their respective Baseline and DAE counterparts, they are superior in learning topological connections, which can be generalized to inter-pixel relationships in related datasets. These models also allow for specification of their exact codebook size, as they are directly trained to utilize the entire set of codes. For topology optimization or related applications, we therefore recommend starting with this variation of the VQGAN, as it only requires a small change to the Baseline codebook. One may then experiment with the codebook dimensions (*i.e.*, the number of embeddings and embedding dimension) to find the proper balance of reconstruction accuracy, learned topology, and model size.

4.2.2 Pixel Swapping. Aside from interpolation, we also conduct a topological robustness test, which we term "pixel swapping", to further distinguish the VQGAN from continuous architectures. From the codebook layout described in Section 3.2.1, we specify two pixels (h_1, w_1) and (h_2, w_2) within $z_q \in \mathbb{R}^{h \times w \times n_z}$ and switch their n_z values (*i.e.*, their latent codes). We repeat the process a fixed number of times, specifying an identical set of swaps for each model to ensure consistency. Using this method, we produce 64 random pixel swaps in the 16×16 latent space, followed by the same quantization and decoding process used for interpolation (or simply decoding for the AE). The image-space equivalent of this operation would be taking two 16×16 chunks of the heat sink and swapping them, since we are evaluating 256×256 images. Clearly, this naive approach has no topological robustness, as it is entirely pixel-based. In contrast, a well-performing latent space would preserve other inter-pixel relationships which it has learned from the training data, such as fluid channel connectivity in the MTO case.

Figure 7 shows the average change in the number of disconnected fluid components after the same 64 swaps are applied to each test design's latent space. In most cases, the original design has only one interconnected fluid component, so the plotted value essentially represents the number of floating fluid "islands" created by the swaps. This is plotted against the original number of solid segments, which correlates to the fluid channel complexity of the heat sink. Generally, the values are highest when few solid segments are present initially—this reflects the behavior of fluid being transferred into a large chunk of solid material, preventing it from connecting to any nearby fluid channels. However, all tested VQGAN models are less prone to this behavior, with a greater proportion of fluid remaining connected to the main flow. A prime example of this effect is shown in Figure 6. Note that the pixel swaps are introduced within a latent space which only models the left half of the heat sink, and are reflected in image space to produce the full sample. Overall, we conclude that the learned topological behavior of each VQGAN is similar in the context of randomized material placements within the latent space, but quite different from (and superior to) that of the continuous AE. Future work may focus on reproducing this behavior in a less random manner.

4.2.3 Additional Latent Space Characteristics. To further uncover possible reasons for the differing latent qualities of

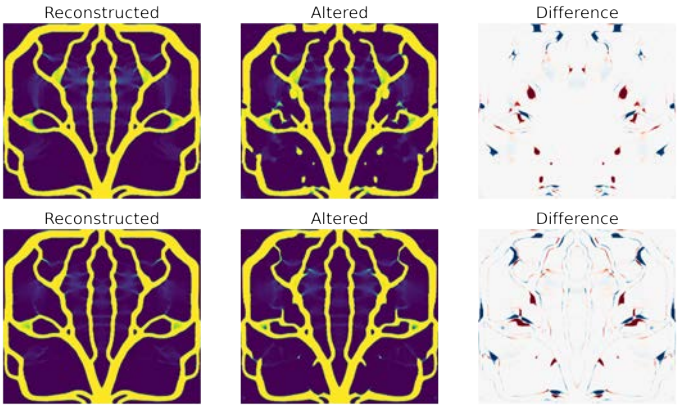

FIGURE 6: TOP ROW: AN EXAMPLE OF AN AE-RECONSTRUCTED SAMPLE AFTER 16 LATENT PIXEL SWAPS. BOTTOM ROW: THE SAME 16 SWAPS DONE IN THE BASELINE VQGAN LATENT SPACE.

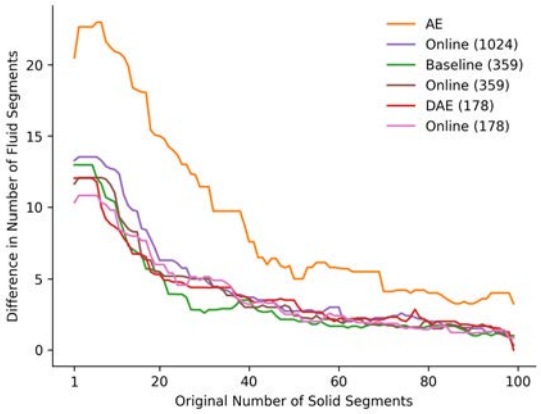

FIGURE 7: AVERAGE CHANGE IN NUMBER OF DISCONNECTED FLUID COMPONENTS AFTER 64 RANDOM LATENT PIXEL SWAPS.

each tested VQGAN, we examine the distribution of latent codes more directly. Figure 8 compares the counts of most common codebook indices between the DAE-VQGAN and Online-178 model, which both feature the same latent size but differ greatly in learned topology, as discussed above. Similar to the results in [45], we find that the DAE-VQGAN reduces the redundancy of certain codes, instead favoring a more balanced distribution. In fact, the DAE-VQGAN featured by far the most balanced codes of any model, including the Baseline VQGAN. Meanwhile, a single index makes up over 7 percent of the Online-178 codebook. In the context of MTO and related datasets, the simpler representation of the DAE-VQGAN does not appear to yield any clear advantage, while the Online-VQGAN better retains topological features when the latent space is altered. It is possible that for highly structured binary images, the redundancy of codes is actually useful—many similar features often exist within a heat sink design, both in terms of pixel value and topology. However, such a claim is not easily proven, and thus we recommend that future work further explores the effects of these discrete code distributions.

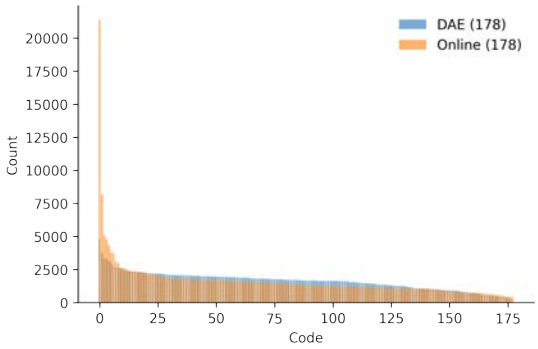

FIGURE 8: A COMPARISON OF ORDERED LATENT CODE DISTRIBUTION FOR THE ONLINE AND DAE MODELS WITH 178 UNIQUE LATENT CODES.

4.3 Generated Heat Sink Characteristics

While the latent space characteristics and reconstruction performance of the VQGAN and AE are important, they do not address the main use case of the MTO dataset: novel output generation. Thus, the transformer portion of the VQGAN (Stage 2) is essential to produce new heat sinks given a desired set of boundary conditions. In this section, we observe the downstream impacts of the Stage 1 models in training and evaluation of the transformer. For this purpose, we use a lightweight version of GPT-2 [54], implemented within a framework called nanoGPT[5].

For additional comparison to continuous architectures, we implement a Deep Convolutional GAN (DCGAN) [55] trained on the previous AE latent space. We also train another DCGAN directly on the VQGAN codebook. Each DCGAN is composed of a near-symmetrical Generator and Discriminator, both with a hidden dimension of 128. The Generator uses blocks composed of a 2D transposed convolutional layer, followed by GroupNorm [48] and a ReLU activation function. The Discriminator is similar, but with 2D convolutional layers and LeakyReLU activation. The network is trained by generating a 13-dimensional normally distributed noise vector z, which is concatenated with the 3-dimensional set of boundary conditions c before being passed to the Generator. Thus, both the Generator input and output are 16-dimensional, since the output also needs to match the Stage 1 model latent dimension. The output is then fed to the Discriminator, alongside the batch of real data, to produce the resulting Binary Cross-Entropy (BCE) loss. In addition, a reconstruction and perceptual loss are calculated for the Generator, similar to the process in Stage 1. Finally, to promote more topologically compliant designs, we implement an L1 loss between the volume fractions of the output and real samples. We scale each loss function such that their magnitudes remain similar during training.

We mainly follow the procedures of [22] regarding the transformer, with the only major change being the addition of the C-VQGAN described previously. We train a new transformer using each of the previous VQGAN Stage 1 models for 500 epochs with learning rate 1e−3 and batch size 16, using the smallest GPT-2 variant with a dropout rate of 0.3. During training, we observe a clear trend with regard to the original codebook size: the larger the codebook, the more difficulty our transformer has in minimizing the loss quickly. This makes sense given the transformer needs to predict a larger range of tokens correctly, and reinforces the argument for maximizing codebook compression in Stage 1. Among the models of equal codebook size, the Baseline and Online-359 models have roughly the same training curves, indicating a similar distribution of codes between them. However, the DAE suffers from a slower training speed compared to the Online-178 model, likely due to its more balanced distribution of codes as seen in Section 4.2.3. While this presents another potential benefit of using a more redundant codebook, we must inspect the quality of transformer outputs to make a true judgment on whether the Online model is superior to the DAE-VQGAN in the context of the MTO dataset.

Once each transformer is trained, we feed in the conditional tokens representing the boundary conditions of the test set. To present more consistent output comparisons, we set $k = 1$ within the Top-K sampling algorithm—so that only the most probable token is chosen at any given step of the transformer's autoregressive generation sequence. However, practical use cases may raise this value to allow for some variation in outputs. In either case, we then obtain the final set of generated designs for each model. Figure 9 shows examples of designs with the best thermal performance. While these generated heat sinks already perform well on their own, we wish to eliminate the presence of intermediate material in them. We thus conduct a small warm-starting procedure to demonstrate how these designs could be post-processed in a real application.

4.4 Warm-Start Optimization And Thermal Analysis of Generated Samples

After obtaining just over 1100 generated heat sink samples from each transformer, we use our MTO solver to run 20 additional adjoint optimization steps on each design. Given the small amount of steps, the main effect of this warm-starting is to drive the pixel values closer to 0 or 1, slightly increasing the final fluid volume fraction in most cases. The warm-start can be seen as a minor post-processing step that takes significantly less time than a full cold-start optimization (*i.e.*, the process that was used to create the initial MTO dataset). Furthermore, the MTO solver provides the thermal performance metrics of each final heat sink.

Before analyzing the thermal performance, we examine the distributions of generated heat sinks and related statistical metrics. We measure the maximum mean discrepancy (MMD) to estimate a distance between each generated distribution and the MTO distribution [56]. We also measure the Relative Diversity, or R-Div, which directly compares the diversity ratio between model outputs and the training data [57]:

$$\text{R-Div} = \frac{\text{trace}(\text{cov}[\mathbf{X}_g, \mathbf{X}_g])}{\text{trace}(\text{cov}[\mathbf{X}_{MTO}, \mathbf{X}_{MTO}])} \tag{4}$$

Next, a topological comparison is performed to gauge the differences in number of connected fluid and solid components in each set of heat sinks. For example, we expect that any viable generated design will (with rare exceptions) only have one

[5] https://github.com/karpathy/nanoGPT

interconnected fluid channel, and a similar number of solid segments to its MTO counterpart. Specifically, we measure Volume Fraction Error (VFE), Solid Segment Error (SSE), and Number of Disconnected Fluid Segments (NDFS)—see Appendix B for an overview of these metrics. Our results show that VQGAN generally produces designs with a similar number of interconnected solid components compared to the MTO data and few disconnected fluid components.

With the statistical and topological measurements complete, we may now observe each sample's thermal performance. Figure 10 provides an example distribution of designs plotted with respect to their power dissipation and mean temperature, and colored by their respective fluid volume fraction (VF). There exists a tendency for heat sinks of higher VF to perform better with regard to these two metrics, and many of the best points (i.e. those along the Pareto front) have VF > 0.6. This makes sense because a higher VF generally corresponds to a larger number of solid bodies, which have a better surface area-to-volume ratio and thus allow for more efficient heat transfer. Future work may focus on an expanded set of heat sinks for which the volume fraction is narrowed to this better-performing range.

Building on the thermal metric plots, we compare their resulting Pareto fronts to the MTO test data Pareto front. For a two-dimensional objective space—formed by the minimization of mean temperature and fluid power dissipation—let $P_I = (PD_I, T_I)$ be the Ideal Point and $P_N = (PD_N, T_N)$ the Nadir Point of the Pareto front. PD_I is the lowest power dissipation of any point and T_I is the lowest mean temperature; the opposite is true for PD_N and T_N. Also let P_G be an arbitrary lower bound on P_I called the "Good Point", and let P_B be an upper bound on P_N called the "Bad Point" [58]. In the MTO case, we set $P_G = (0, 0)$ and $P_B = (75, 50)$ as the fixed points of reference, based on the typical performance values observed. To allow for numerically consistent metrics across different models and datasets, we then scale the Pareto points along each of the objective dimensions, such that P_G becomes $(0, 0)$ and P_B becomes $(1, 1)$, with the entire objective area scaled to 1.

Figure 11 shows the Pareto fronts for each set of heat sinks. Notably, the DAE-VQGAN Pareto front features several points which *dominate* all other curves—that is, they perform better in terms of both power dissipation and mean temperature. We measure this dominance by calculating the Pareto-Dominated Area (PDA), which roughly gauges the distance of the Pareto front from P_B. This metric is simply the union of rectangular areas between each Pareto point and P_B, which extend upward and to the right (given we are minimizing each objective). We refer to PDA measured in the scaled Pareto space as PDA_s, and also define the Non-Pareto-Dominated Area as $1 - \text{PDA}_s$ to produce a metric which may be thought of as a distance to P_G. An optimal Pareto front would thus yield a value of zero for this metric.

We observe that based on $1 - \text{PDA}_s$, VQGAN generally attains a Pareto front which outperforms that of not only the DCGAN, but also the MTO test data itself. Looking beyond this at the distribution of all points, we also find that the DAE-VQGAN features by far the highest power dissipation variance of any VQGAN. This yields many samples with higher J than desired. However, it also produces the best designs of any model for a select set of points with $J < 30\underline{J_1}$. In contrast, we find that all VQGANs produce similar distributions to the test data with respect to mean temperature. Overall, we conclude that each VQGAN produces a highly diverse set of generated designs compared to the test data. Many of these are not only valid designs, but feature near-optimal thermal performance as well.

Table 3 summarizes the overall results of the transformers and DCGANs, from which several interesting observations can be made. For statistical metrics, the DCGAN models yield higher MMD (Gaussian kernel with $\sigma = 2$) and significantly lower R-Div values compared to the transformers. From this we hypothesize that mode collapse is a major issue, even when the latent space is compressed via quantization (as in DCGAN-VQ). We confirm this problem is present in generated designs of similar volume fraction: often each DCGAN produces the exact same output for slightly different boundary conditions, with Figure 9 illustrating an example of this. Upon another inspection of Figure 11, we observe this behavior also translates to a more restricted and suboptimal set of Pareto points. Nonetheless, training the DCGAN on the Baseline VQGAN codebook provides a significant performance boost, showing the benefits of a compressed discrete latent space in generative applications. We find that the DAE-VQGAN-based transformer achieves the best overall performance, both in terms of topological and thermal characteristics. Interestingly, while the smaller codebooks were easier for the transformer to learn, they did not necessarily yield better results (as in the case of the Online-178 transformer). We can see that below a certain codebook size, the output diversity as measured by R-Div begins to suffer, although the DAE-VQGAN appears significantly more robust to this issue compared to its Online counterpart. We thus infer that the distribution of latent codes and decoder quality play a large role in the performance of transformer outputs.

Given that the DAE-VQGAN was the only Stage 1 model featuring a significant difference in latent space composition compared to the Baseline, this result provides a possible argument for using the DAE in design generation tasks. However, additional work needs to be done to clarify exactly how VQGAN latent space properties impact transformer outputs. We also note that the Pareto fronts generated are fairly sparse, given a test set of just over 1100 total samples. A larger sample size would thus be preferable to confirm our results. Because the DAE-VQGAN performed worse in latent space tests, the degree of its benefits in the generative context also remains unclear. However, one takeaway is that all VQGAN-based transformers performed quite well from the statistical standpoint (*i.e.*, they yielded low MMD and high R-Div) and are highly capable of producing new, thermally effective designs. This is despite the fact that we used the smallest GPT-2 model configuration available in nanoGPT. Furthermore, we have shown that a large degree of data compression is possible for binary or grey-scale image datasets such as the MTO dataset—which are quite repetitive from the pixel-based standpoint—yet may feature high levels of topological complexity or other patterns. Overall, we believe the Online-VQGAN presents the most practical approach to modeling such datasets due to its fully controllable codebook size and desirable latent properties.

TABLE 3: PERFORMANCE METRICS FOR EACH TRANSFORMER AND DCGAN, BASED ON THE STAGE 1 MODEL IT WAS TRAINED WITH.

Model (Codebook Size)	Training Log-Loss ↓	MMD ($\sigma = 2$) ↓	R-Div ↑	VFE (%) ↓	SSE (%) ↓	NDFS ↓	$1 - \text{PDA}_s$ ↓
DCGAN-AE (n/a)	n/a	0.0621	0.550	1.13	52.6	5.532	0.518
DCGAN-VQ (359)	n/a	0.0489	0.652	1.30	27.5	4.782	0.422
Online (1024)	0.209	0.0423	1.003	2.64	29.2	**0.992**	0.345
Baseline (359)	-0.397	0.0426	**1.005**	2.59	30.9	0.995	0.343
Online (359)	-0.452	0.0422	0.991	2.48	28.6	1.097	0.334
DAE (178)	-0.709	0.0422	0.976	**1.47**	**22.9**	1.108	**0.310**
Online (178)	**-0.989**	**0.0421**	0.912	3.04	30.5	1.004	0.343

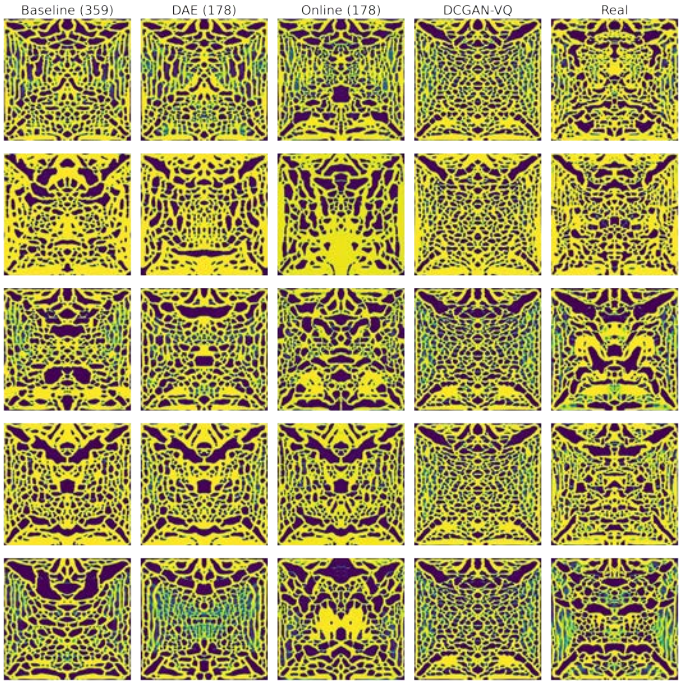

FIGURE 9: SAMPLE OUTPUTS FEATURING THE BEST AVERAGE THERMAL PERFORMANCE ACROSS EACH TRANSFORMER. NOTE THAT THE VISIBLE INTERMEDIATE VALUES ARE LARGELY RESOLVED DURING THE WARM-START PROCESS.

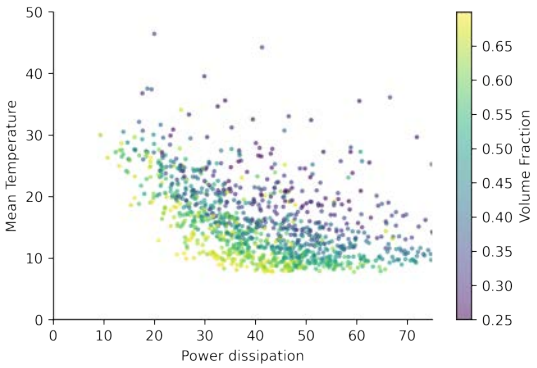

FIGURE 10: THE THERMAL PERFORMANCE OF DESIGNS GENERATED BY THE TRANSFORMER TRAINED WITH THE DAE-VQGAN MODEL, WITH POINTS COLORED BY THEIR RESPECTIVE VF.

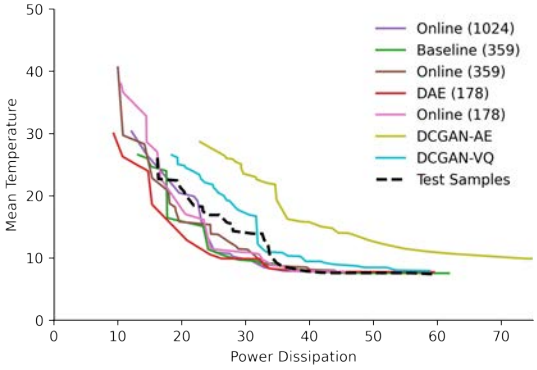

FIGURE 11: THE PARETO FRONTIER FOR EACH TRANSFORMER'S SET OF GENERATED SAMPLES ON THE TEST SET.

5. CONCLUSION

In this paper, we introduced a new, two-dimensional dataset for Multiphysics Topology Optimization (MTO) which presents a wide range of optimized heat sinks with varying volume fraction, power dissipation, and Reynolds number. We also proposed several statistical, topological, and thermal metrics to thoroughly gauge the performance of machine learning models on the MTO dataset, both in terms of image reconstruction and new design generation. Importantly, our approach can also be used for other two-dimensional datasets featuring grey-scale or binary images. Building on this foundation, we presented the Vector-Quantized GAN (VQGAN) [22] as a practical solution in heat sink design applications—which often utilize slow and computationally expensive physics-based models to predict optimal designs. Through a detailed latent space analysis with comparisons to continuous models such as Autoencoders (AEs) and Deep Convolutional GANs (DCGANs), we found VQGAN provides a superior latent space model and aids in generative design applications.

Moreover, we implemented two recently-proposed augmentations to the VQGAN architecture: the Decoupled Autoencoder VQGAN (DAE-VQGAN) [45] and VQGAN with Online Clustered Codebook (Online-VQGAN) [53]. We found benefits in using either model over the Baseline. Firstly, the Online model guarantees 100% codebook usage by pushing unused codebook

vectors closer to encoded features and offers equal or better performance compared to a Baseline VQGAN of the same latent size. In the MTO context, it also provides more convincing interpolation between samples by preserving a high level of topological connectivity compared to the other models tested. Secondly, the DAE-VQGAN balances the latent representation via its decoupled training process—which emphasizes codebook quality during the first stage and reconstruction quality during the second stage. The DAE model demonstrates that the MTO dataset can be reconstructed effectively with just 178 codes given a 16×16 latent feature map, preserving all major fluid channel connections in the test samples. The main benefits of the DAE-VQGAN are observed in the generative context, as a transformer trained on its latent space produces more thermally diverse designs while better retaining the topological characteristics of the original MTO data. We thus encourage further research of the Online-VQGAN and DAE-VQGAN for related engineering and design applications.

Despite these promising findings, some limitations exist in this paper and will be covered in future work. Firstly, we do not fully address the relationship between latent properties discussed in Section 4.2 and the downstream performance of the transformers and DCGANs tested. Additional work should be done to understand what precise representations the transformer is able to learn from the tokenized inputs it is trained on. It would also be helpful to compare our models with additional state-of-the-art frameworks such as Diffusion models [59] and more recent deep GAN architectures that better mitigate mode collapse—both as a direct measure of their performance and to further validate the MTO dataset as an effective benchmark. However, we would also need to address some minor problems with the current MTO data before implementing these new models. In this paper, we made limited comparisons to the thermal performance of the MTO data itself due to an issue with power dissipation constraint satisfaction during cold-start optimization. We are working to regenerate the affected samples so they fully satisfy all constraints, and will make the dataset publicly available once this is done. Nonetheless, we remain confident that our results will be mostly unaffected by this issue, given the high diversity and strong thermal performance of current generated samples. Finally, future work will thus expand this paper's experiments to a three-dimensional context as well.

ACKNOWLEDGEMENTS

We acknowledge the support from the National Science Foundation through award #1943699 and from the U.S. Department of Energy's Advanced Research Projects Agency-Energy (ARPA-E) DIFFERENTIATE program through award DE-AR0001216. We also acknowledge the University of Maryland supercomputing resources (http://hpcc.umd.edu) made available for conducting the research reported in this paper. We also thank Katie Kirsch and Ram Ranjan at Raytheon Technologies for helpful discussions on appropriate boundary conditions and constraints for the MTO problem formulation.

REFERENCES

[1] Molesky, Sean, Lin, Zin, Piggott, Alexander Y, Jin, Weiliang, Vucković, Jelena and Rodriguez, Alejandro W. "Inverse design in nanophotonics." *Nature Photonics* Vol. 12 No. 11 (2018): pp. 659–670.

[2] Zunger, Alex. "Inverse design in search of materials with target functionalities." *Nature Reviews Chemistry* Vol. 2 No. 4 (2018): p. 0121.

[3] Lu, Lu, Pestourie, Raphael, Yao, Wenjie, Wang, Zhicheng, Verdugo, Francesc and Johnson, Steven G. "Physics-informed neural networks with hard constraints for inverse design." *SIAM Journal on Scientific Computing* Vol. 43 No. 6 (2021): pp. B1105–B1132.

[4] Yang, Sunwoong, Lee, Sanga and Yee, Kwanjung. "Inverse design optimization framework via a two-step deep learning approach: application to a wind turbine airfoil." *Engineering with Computers* Vol. 39 No. 3 (2023): pp. 2239–2255.

[5] Kingma, Diederik P and Welling, Max. "Auto-encoding variational bayes." *arXiv preprint arXiv:1312.6114* (2013).

[6] Burda, Yuri, Grosse, Roger and Salakhutdinov, Ruslan. "Importance weighted autoencoders." *arXiv preprint arXiv:1509.00519* (2015).

[7] Gulrajani, Ishaan, Kumar, Kundan, Ahmed, Faruk, Taiga, Adrien Ali, Visin, Francesco, Vazquez, David and Courville, Aaron. "Pixelvae: A latent variable model for natural images." *arXiv preprint arXiv:1611.05013* (2016).

[8] Cremer, Chris, Li, Xuechen and Duvenaud, David. "Inference suboptimality in variational autoencoders." *International Conference on Machine Learning*: pp. 1078–1086. 2018. PMLR.

[9] Fortuin, Vincent, Baranchuk, Dmitry, Raetsch, Gunnar and Mandt, Stephan. "GP-VAE: Deep Probabilistic Time Series Imputation." Chiappa, Silvia and Calandra, Roberto (eds.). *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, Vol. 108: pp. 1651–1661. 2020. PMLR. URL https://proceedings.mlr.press/v108/fortuin20a.html.

[10] Goodfellow, Ian J., Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron and Bengio, Yoshua. "Generative Adversarial Networks." (2014). URL 1406.2661.

[11] Arjovsky, Martin, Chintala, Soumith and Bottou, Léon. "Wasserstein GAN." (2017). URL 1701.07875.

[12] Gulrajani, Ishaan, Ahmed, Faruk, Arjovsky, Martin, Dumoulin, Vincent and Courville, Aaron C. "Improved training of wasserstein gans." *Advances in neural information processing systems* Vol. 30 (2017).

[13] Isola, Phillip, Zhu, Jun-Yan, Zhou, Tinghui and Efros, Alexei A. "Image-to-image translation with conditional adversarial networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*: pp. 1125–1134. 2017.

[14] Mao, Xudong, Li, Qing, Xie, Haoran, Lau, Raymond YK, Wang, Zhen and Paul Smolley, Stephen. "Least squares generative adversarial networks." *Proceedings of the IEEE international conference on computer vision*: pp. 2794–2802. 2017.

[15] Nichol, Alexander Quinn and Dhariwal, Prafulla. "Improved denoising diffusion probabilistic models." *Interna-*

tional Conference on Machine Learning: pp. 8162–8171. 2021. PMLR.

[16] Croitoru, Florinel-Alin, Hondru, Vlad, Ionescu, Radu Tudor and Shah, Mubarak. "Diffusion models in vision: A survey." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[17] Wang, Zhendong, Jiang, Yifan, Zheng, Huangjie, Wang, Peihao, He, Pengcheng, Wang, Zhangyang, Chen, Weizhu, Zhou, Mingyuan et al. "Patch diffusion: Faster and more data-efficient training of diffusion models." *Advances in Neural Information Processing Systems* Vol. 36 (2024).

[18] Bendsoe, Martin Philip and Sigmund, Ole. *Topology optimization: theory, methods, and applications*. Springer Science & Business Media (2013).

[19] Behzadi, Mohammad Mahdi and Ilies, Horea T. "On the Connectedness of the Topology Optimization Predictors." *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Vol. 87301: p. V03AT03A024. 2023. American Society of Mechanical Engineers.

[20] Behzadi, Mohammad Mahdi, Chen, Jiangce and Ilies, Horea T. "Taming Connectedness in Machine-Learning-Based Topology Optimization with Connectivity Graphs." *Computer-Aided Design* Vol. 168 (2024): p. 103634.

[21] Vaswani, Ashish, Shazeer, Noam, Parmar, Niki, Uszkoreit, Jakob, Jones, Llion, Gomez, Aidan N, Kaiser, Łukasz and Polosukhin, Illia. "Attention is all you need." *Advances in neural information processing systems* Vol. 30 (2017).

[22] Esser, Patrick, Rombach, Robin and Ommer, Björn. "Taming Transformers for High-Resolution Image Synthesis." (2021). URL 2012.09841.

[23] Ahmed, Hamdi E, Salman, BH, Kherbeet, A Sh and Ahmed, MI. "Optimization of thermal design of heat sinks: A review." *International Journal of Heat and Mass Transfer* Vol. 118 (2018): pp. 129–153.

[24] Bendsøe, Martin P. "Optimal shape design as a material distribution problem." *Structural optimization* Vol. 1 (1989): pp. 193–202.

[25] Sigmund, Ole. "Design of multiphysics actuators using topology optimization–Part I: One-material structures." *Computer methods in applied mechanics and engineering* Vol. 190 No. 49-50 (2001): pp. 6577–6604.

[26] Dede, Ercan M. "Multiphysics topology optimization of heat transfer and fluid flow systems." *proceedings of the COMSOL Users Conference*, Vol. 715. 2009.

[27] Koga, Adriano A., Lopes, Edson Comini C., Villa Nova, Helcio F., de Lima, Cícero R. and Silva, Emílio Carlos Nelli. "Development of heat sink device by using topology optimization." *International Journal of Heat and Mass Transfer* Vol. 64 (2013): pp. 759–772. DOI https://doi.org/10.1016/j.ijheatmasstransfer.2013.05.007. URL https://www.sciencedirect.com/science/article/pii/S0017931013003979.

[28] Dede, Ercan M., Joshi, Shailesh N. and Zhou, Feng. "Topology Optimization, Additive Layer Manufacturing, and Experimental Testing of an Air-Cooled Heat Sink." *Journal of Mechanical Design* Vol. 137 No. 11 (2015). DOI 10.1115/1.4030989. URL https://asmedigitalcollection.asme.org/mechanicaldesign/article-pdf/137/11/111403/6800273/md_137_11_111403.pdf, URL https://doi.org/10.1115/1.4030989. 111403.

[29] Haertel, Jan H.K., Engelbrecht, Kurt, Lazarov, Boyan S. and Sigmund, Ole. "Topology optimization of a pseudo 3D thermofluid heat sink model." *International Journal of Heat and Mass Transfer* Vol. 121 (2018): pp. 1073–1088. DOI https://doi.org/10.1016/j.ijheatmasstransfer.2018.01.078. URL https://www.sciencedirect.com/science/article/pii/S0017931017337146.

[30] Liang, Xuan, Li, Angran, Rollett, Anthony D and Zhang, Yongjie Jessica. "An isogeometric analysis-based topology optimization framework for 2D cross-flow heat exchangers with manufacturability constraints." *Engineering with Computers* (2022): pp. 1–24.

[31] Xie, Liyao, Zhang, Yan, Ge, Minghui and Zhao, Yulong. "Topology optimization of heat sink based on variable density method." *Energy Reports* Vol. 8 (2022): pp. 718–726.

[32] Fedorov, Andrei G and Viskanta, Raymond. "Three-dimensional conjugate heat transfer in the microchannel heat sink for electronic packaging." *International Journal of Heat and Mass Transfer* Vol. 43 No. 3 (2000): pp. 399–415.

[33] Parrott, Corey M, Abueidda, Diab W and James, Kai A. "Multi-Head Self-Attention Generative Adversarial Networks for Multiphysics Topology Optimization." *AIAA Journal* Vol. 61 No. 2 (2023): pp. 726–738.

[34] Gray, Robert. "Vector quantization." *IEEE Assp Magazine* Vol. 1 No. 2 (1984): pp. 4–29.

[35] Kohonen, Teuvo and Kohonen, Teuvo. "Learning vector quantization." *Self-organizing maps* (1995): pp. 175–189.

[36] Wu, Ze-bin and Yu, Jun-qing. "Vector quantization: a review." *Frontiers of Information Technology & Electronic Engineering* Vol. 20 No. 4 (2019): pp. 507–524.

[37] van den Oord, Aaron, Vinyals, Oriol and Kavukcuoglu, Koray. "Neural Discrete Representation Learning." (2018). URL 1711.00937.

[38] Razavi, Ali, van den Oord, Aaron and Vinyals, Oriol. "Generating Diverse High-Fidelity Images with VQ-VAE-2." (2019). URL 1906.00446.

[39] Van den Oord, Aaron, Kalchbrenner, Nal, Espeholt, Lasse, Vinyals, Oriol, Graves, Alex et al. "Conditional image generation with pixelcnn decoders." *Advances in neural information processing systems* Vol. 29 (2016).

[40] Yu, Jiahui, Li, Xin, Koh, Jing Yu, Zhang, Han, Pang, Ruoming, Qin, James, Ku, Alexander, Xu, Yuanzhong, Baldridge, Jason and Wu, Yonghui. "Vector-quantized image modeling with improved vqgan." *arXiv preprint arXiv:2110.04627* (2021).

[41] Zhu, Zixin, Feng, Xuelu, Chen, Dongdong, Bao, Jianmin, Wang, Le, Chen, Yinpeng, Yuan, Lu and Hua, Gang. "Designing a Better Asymmetric VQGAN for StableDiffusion." *arXiv preprint arXiv:2306.04632* (2023).

[42] Gu, Shuyang, Chen, Dong, Bao, Jianmin, Wen, Fang, Zhang, Bo, Chen, Dongdong, Yuan, Lu and Guo, Baining.

"Vector quantized diffusion model for text-to-image synthesis." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*: pp. 10696–10706. 2022.

[43] Song, Binyang, Zhou, Rui and Ahmed, Faez. "Multi-modal machine learning in engineering design: A review and future directions." *Journal of Computing and Information Science in Engineering* Vol. 24 No. 1 (2024): p. 010801.

[44] Parrott, Corey M, Abueidda, Diab W and James, Kai A. "Multidisciplinary Topology Optimization Using Generative Adversarial Networks for Physics-Based Design Enhancement." *Journal of Mechanical Design* Vol. 145 No. 6 (2023): p. 061704.

[45] Hu, Tianyang, Chen, Fei, Wang, Haonan, Li, Jiawei, Wang, Wenjia, Sun, Jiacheng and Li, Zhenguo. "Complexity Matters: Rethinking the Latent Space for Generative Modeling." *arXiv preprint arXiv:2307.08283* (2023).

[46] Yu, Minghao, Ruan, Shilun, Gu, Junfeng, Ren, Mengke, Li, Zheng, Wang, Xinyu and Shen, Changyu. "Three-dimensional topology optimization of thermal-fluid-structural problems for cooling system design." *Structural and Multidisciplinary Optimization* Vol. 62 No. 6 (2020): pp. 3347–3366.

[47] McKay, Michael D, Beckman, Richard J and Conover, William J. "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code." *Technometrics* Vol. 42 No. 1 (2000): pp. 55–61.

[48] Wu, Yuxin and He, Kaiming. "Group Normalization." *CoRR* Vol. abs/1803.08494 (2018). URL 1803.08494, URL http://arxiv.org/abs/1803.08494.

[49] Dosovitskiy, Alexey and Brox, Thomas. "Generating images with perceptual similarity metrics based on deep networks." *Advances in neural information processing systems* Vol. 29 (2016).

[50] Johnson, Justin, Alahi, Alexandre and Fei-Fei, Li. "Perceptual losses for real-time style transfer and super-resolution." *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*: pp. 694–711. 2016. Springer.

[51] Kingma, Diederik P and Ba, Jimmy. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).

[52] Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya and Salakhutdinov, Ruslan. "Dropout: a simple way to prevent neural networks from overfitting." *The journal of machine learning research* Vol. 15 No. 1 (2014): pp. 1929–1958.

[53] Zheng, Chuanxia and Vedaldi, Andrea. "Online clustered codebook." *Proceedings of the IEEE/CVF International Conference on Computer Vision*: pp. 22798–22807. 2023.

[54] Radford, Alec, Wu, Jeffrey, Child, Rewon, Luan, David, Amodei, Dario, Sutskever, Ilya et al. "Language models are unsupervised multitask learners." *OpenAI blog* Vol. 1 No. 8 (2019): p. 9.

[55] Radford, Alec, Metz, Luke and Chintala, Soumith. "Unsupervised representation learning with deep convolutional generative adversarial networks." *arXiv preprint arXiv:1511.06434* (2015).

[56] Borgwardt, Karsten M, Gretton, Arthur, Rasch, Malte J, Kriegel, Hans-Peter, Schölkopf, Bernhard and Smola, Alex J. "Integrating structured biological data by kernel maximum mean discrepancy." *Bioinformatics* Vol. 22 No. 14 (2006): pp. e49–e57.

[57] Chen, Wei and Fuge, Mark. "Synthesizing designs with interpart dependencies using hierarchical generative adversarial networks." *Journal of Mechanical Design* Vol. 141 No. 11 (2019): p. 111403.

[58] Wu, Jin and Azarm, Shapour. "Metrics for quality assessment of a multiobjective design optimization solution set." *J. Mech. Des.* Vol. 123 No. 1 (2001): pp. 18–25.

[59] Rombach, Robin, Blattmann, Andreas, Lorenz, Dominik, Esser, Patrick and Ommer, Björn. "High-resolution image synthesis with latent diffusion models." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*: pp. 10684–10695. 2022.

APPENDIX A. GOVERNING EQUATIONS FOR MTO

Power Dissipation

In order to prevent unrealistic designs, the power dissipation of the fluid device is considered a constraint:

$$J = -\int_{\Gamma_{in\cup out}} (p + \frac{1}{2}\mathbf{u}\cdot\mathbf{u})\mathbf{u}\cdot\mathbf{n}\mathrm{d}\Gamma \tag{5}$$

where J denotes the power dissipation and Γ is the boundary.

Governing Equations

The thermal-fluid MTO problem is modeled by steady-state incompressible Navier-Stokes equations and energy balance equations. The two sets of governing equations are presented below.

Navier-Stokes equations:

$$\begin{cases} -\nabla\cdot\mathbf{u} = 0 & \text{in } \Omega \\ \rho(\mathbf{u}\cdot\nabla)\mathbf{u} - \eta\nabla\cdot(\nabla\mathbf{u} + \nabla\mathbf{u}^T) + \nabla p - \mathbf{F}_b = 0 & \text{in } \Omega \\ \mathbf{u} = \mathbf{u}_D & \text{on } \Gamma_D^F \\ [\eta(\nabla\mathbf{u} + \nabla\mathbf{u}^T) - p\mathbf{I}]\cdot\mathbf{n} = \mathbf{g}_N & \text{on } \Gamma_N^F \end{cases} \tag{6}$$

Energy conservation equations:

$$\begin{cases} \rho c(\mathbf{u}\cdot\nabla T) = \nabla\cdot(k\nabla T) + Q & \text{in } \Omega \\ T = T_D & \text{on } \Gamma_D^T \\ k\nabla T\cdot\mathbf{n} = q_N & \text{on } \Gamma_N^T \end{cases} \tag{7}$$

where Ω denotes the computational domain, Γ is the boundary, the subscripts D and N represent the Dirichlet and Neumann parts of the boundary, and the superscripts F and T represent the fluid and thermal mechanics problems, respectively. ρ is the fluid mass density, and $\mathbf{u}$ and p are the velocity and pressure fields, respectively. η is the dynamic viscosity, while $\mathbf{F}_b$ is the body force of the fluid. $\mathbf{u}_D$ and $\mathbf{g}_N$ are the specified velocity and stress distributions on the boundaries Γ_D^F and Γ_N^F, respectively. T is the temperature field, $\mathbf{n}$ is the unitary outward normal, c is the specific heat capacity, k is the thermal conductivity, Q is the heat source capacity, and T_D and q_N are the specified temperature and heat flux on the boundaries Γ_D^T and Γ_N^T, respectively.

APPENDIX B. TOPOLOGICAL COMPARISON METRICS

We define several intuitive metrics for quantifying the topological differences between a generated or reconstructed design and its real counterpart. Note that any grey-scale designs should be binarized before such an analysis. Let N_{solid} be the number of disconnected solid segments in a design and N_{fluid} the number of disconnected fluid segments. Also let VF be the fluid volume fraction of a design. For MTO, we desire $N_{fluid} = 1$ in all designs, as any other disconnected fluid segments would amount to a waste of material. The Number of Disconnected Fluid Segments (NDFS) is thus simply:

$$NDFS = N_{fluid,gen} \tag{8}$$

No difference is calculated with respect to the real design because a clear minimization objective exists. For more general cases, the Fluid Segment Error (FSE) may be determined as:

$$FSE = \frac{|N_{fluid,gen} - N_{fluid,real}|}{N_{fluid,real}} \tag{9}$$

Similarly, the Solid Segment Error (SSE) is:

$$SSE = \frac{|N_{solid,gen} - N_{solid,real}|}{N_{solid,real}} \tag{10}$$

In short, SSE and FSE correspond to the Betti errors for the respective binary regions. The fact that they represent fluid or solid is an arbitrary specification for our MTO data. Finally, the volume fraction error (VFE) is also straightforward:

$$VFE = |\mathrm{VF}_{gen} - \mathrm{VF}_{real}| \tag{11}$$