deepSPACE: Generative AI for Configuration Design Space Exploration

Emilio M. Botero* and Jordan T. Smart*
Stargazer Design Technologies INC

This paper introduces deepSPACE, a generalized design space exploration method. deepSPACE employs artificial intelligence to collate dissimilar solutions into a unified design space. Addressing the limitations of traditional manual processes in configuration design, such as brainstorming, deepSPACE enables true design synthesis. The paper critiques the shortcomings of casting design problems as a pure optimization task due to technical complexities and the loss of designer intuition. Built upon a novel generative AI-based mixed-integer constraint solver, deepSPACE addresses these challenges by enabling rapid iterations, exploring design spaces, and providing novel configuration options. deepSPACE generates multiple viable solutions, accompanied by immediate 3D CAD models and performance evaluations. This leaves the designer with a greater understanding and intuition of the design space than ever before. The groundbreaking aspect lies in its ability to navigate the vast design space through a style-driven approach, transforming complex optimization into simplified 1D searches. This paper describes the implementation of and demonstrates its applicability to both physical and operational system design.

I. Introduction

Configuration design is the essence of design. It encompasses the fundamental decisions of component selection and sizing, subject to requirements. Beyond the ever-important aesthetics of a design, performance, and performance margins are essentially fixed upon configuration selection. These fundamental decisions will either help or hinder the long-term success of a design project, potentially for decades.

While the later phases of systems engineering and design are highly formalized, configuration design is largely a freeform manual process as the initial phase of conceptual design. This stage of design has historically been encompassed by things like individual or group brainstorming [1] and TRIZ [2]. These processes are limited by the designer's intuition to identify plausible solutions to the problem at hand. Ultimately, they rely on the "Eureka effect", either in the form of a sudden idea or a slow process.

Intrinsic to these existing methods is sparse human understanding of technical trade-offs, especially in highly coupled multidisciplinary design spaces. Even the best engineering teams typically rely on feel to evaluate the relative performance of different configurations. At best, it results in a Pugh matrix filled with qualitative analysis of many options [3]. Yet, we remain at the whims of inspiration to determine what possible architectures to consider in the first place.

Iterating on previously successful designs is likely to produce results so long as the domain of the new problem is not significantly dissimilar to previous problems. However, this can limit innovation, produce dangerous complacency, and fails outright when approaching problems involving new environments or considering architectures taking advantage of emerging technologies.

The next steps typically have teams down-select from the Pugh method to the most promising configurations for conceptual-level analysis. At this junction, deciding which configurations to down-select is often subjective, resulting in conservative decisions given ever-present time constraints on design. If a design fails to pass the conceptual design phase, they need to return to basic configuration design and try the process again with updated knowledge.

A. Conceptual Design

Conceptual design takes the requirements identified in a problem definition stage and creates a design scheme with a set of analyzed performance. This synthesis stage identifies potential problems, searches for solution principles, creates concept variants, compares them, down-selects, and refines the selected concepts. Configuration design is the core

*Founder

subtask of conceptual design, where concept variants are created, compared, and down-selected. The end result of conceptual design is "frozen" high-level characteristics that define the design. The decisions made in configuration design and conceptual design lock in the value for the remainder of the project.

While performing conceptual design, the designers often return to the problem definition. They constantly challenge the requirements for several reasons, primarily because they may not be fully fixed. Rather, they are negotiating customers' and stakeholders' true needs and wants, driven by tradeoffs and physics. However, knowing these tradeoffs only comes after exploring the design space to find solutions. In essence, conceptual design is not a linear process but rather an iterative negotiation between physics, technical capabilities, regulations, and business cases.

This iteration of the conceptual design process also answers many other aspects, such as what technologies to consider. If this is an ambitious project, then potentially immature technologies will be examined. The selection of technologies leads to analyses. Some simpler established technologies have straightforward evaluation methods, while others are more complex. Along the way, different assumptions can be made based on the technologies and the analyses. Part of the design process is to challenge these assumptions and evaluate the performance not only of the design but also of the analyses themselves.

One goal of the designer during the iteration process is to determine the objective functions of interest to the customer and the problem itself. These objectives may be used in a single or multi-objective optimization problem or as guiding principles through the search process.

B. MINLP Optimization

We can take the practice of configuration design and pose it as an optimization problem. However, the authors argue this is misguided for both technical and operational reasons. Let's begin with the technical issues.

Configuration design encompasses the selection of components, which are categorical variables composed of continuous variables such as dimensions. As an example, if we are designing an airplane, the optimizer must select the wing to be designed and the optimal dimensions of that wing. This leads to necessarily a mixed-integer formulation of both discrete and continuous variables. Common optimization algorithms, such as gradient-based optimizers, are of no use in this regime. This includes many classic Mixed Integer Non-Linear Program (MINLP) solvers, which themselves can exploit gradients of the analyses in subspaces [4]. For many aerospace problems, the existing analyses are black-box. They don't even have gradients available, much less simple analytic functions. When posed this way, as a MINLP black box, global optimization is NP-hard [5]. This leads us to use algorithms such as evolutionary algorithms and genetic algorithms [6]. In practice, solving takes extreme time commitments. Some have tried including AI in the MINLP problem as well[7]. Others have tried surrogate model-based approaches [8] with a type of rounding relaxation with discrete variables.

Typical optimization algorithms also do not easily lend to configuration design due to the variable lengths data [9]. For example, a Boeing 747 is far more complex than a Piper Cub, but a good optimizer for aircraft configuration design should be able to create both. At the crux are coupled discrete and continuous data types of variable sizes. The data space should be able to handle the vast number of parts of a 747 and the diversity of component types between the Piper Cub and a 747.

Technical issues aside, casting the design problem as an optimization problem leaves the designer disempowered. The designer has limited time and thus analysis function evaluations. Using those function evaluations on a MINLP optimization leaves them lacking intuition and understanding. For example, if the objective function changes midway through the process due to customer needs evolving, they start from scratch with nothing to show. The function evaluations typically are not leveraged to enhance future optimization runs. The negotiation and iteration of requirements is impractical using optimization, no real sense of trade offs are acquired. If the requirements are posed to be infeasible, no solutions may arise.

Good aerospace designers typically sweep both the discrete and continuous spaces to build carpet plots [10]. This helps them identify trade-offs in things such as cost vs performance by changing configurations. It leads to some understanding and intuition of the design space and allows for negotiations and iterations towards a viable product. However, this still leaves a gap in exploration, creativity, and optimization. Designers tend to to stay local to things they know and trust, sometimes for good reason and sometimes not.

C. Shortcomings

We can examine current methods of conceptual design and, thus, configuration design in the manner of hypothesis testing[11]. As an analog to the Type I "false positive" error, designers may have one or a small subset of configurations

as the only feasible solutions – appearing to have solved the problem once and for all when they have not. Alternatively, as an analog to the Type II "false negative" error – designers may have no solution or a manifestly underwhelming one despite the underlying analysis or simulation process being perfectly capable of modeling and verifying a much better solution. Worst of all – when presented with the results of an optimization, a designer may have no way of telling whether they're in either situation, as optimizers can only report the results of the optimization process, along with perhaps a history and fitted surrogate.

The current processes leave both the designers and the customers underwhelmed. Designers are not usually fully confident they have selected "THE" design, which will be the most successful and ultimately the most profitable for the company. This results in a fear of missing out: Have their competitors designed something better in response?

The primary concern here for designers is that configuration design and the subsequent conceptual design fundamentally take too long. The process being manually driven slows progress. Response times for RFPs can be as short as 60 days. That leaves designers scrambling to "close" the design. If selected, that RFP locks in the configuration, and the requesting agency expects a design similar to the one proposed. Even outside government contracting, we see time pressures, competitors launching products, investors seeking a return, and regular product cycle upgrades. Every project has limited funds and thus limited time to create a design. All of these issues have pointed towards a need for methods that build intuition, explore the configuration space, perform optimization, assist in negotiating requirements, and finally do this faster than in a manual fashion.

II. deepSPACE

A. Overview

We present deepSPACE, a generative conceptual design methodology to surpass these limitations. deepSPACE was built to solve the configuration design issues and move high-fidelity multi-disciplinary analysis earlier into the design process. deepSPACE enables rapid iterations from relatively few function evaluations and/or use of existing data. It empowers designers to explore the design space, build intuition, understand trade-offs, and provide them with new configuration options that they have never seen before. The key to this approach goes beyond just revolutionary numerical methods that blend AI with traditional MDO; it also includes rapid visualizations. These visualizations help convey a story that can be quickly communicated to customers and stakeholders.

Underlying deepSPACE is a Generative AI-powered mixed-integer constraint solver. This solver learns the joint probability distribution of the design problem connecting specifications, including things like components and dimensions, to their performance characteristics or requirements. From this, we can generate, from requirements, multiple viable configuration solutions. deepSPACE then immediately plots a 3-D CAD model (or 2-D model, depending on the domain) of the configuration along with performance results. deepSPACE can synthesize multiple solutions simultaneously for examination. Users can sweep the requirements space to gather statistics of potential configurations. This naturally lends itself to accelerating optimization processes, however deepSPACE's capabilities extend well beyond that.

B. Configuration Design

We broadly but firmly define configuration design as the process of determining a set of configurations, $\mathbf{C}$, given a set of requirements, $\mathbf{R}$, without regard to any a priori constraint on $\mathbf{R}$. Defined so broadly, we immediately encounter the fact that for most problems, $\mathbf{C}$ cannot be determined simply as a direct relationship or function of $\mathbf{R}$. Trivially, many configurations might satisfy the requirements. Formally, we seek to determine, for a given $c \in \mathbf{C}$, and $r \in \mathbf{R}$, whether $(c, r) \in \mathbf{F}$, where $\mathbf{F}$ is the subset of $\mathbf{C} \times \mathbf{R}$ which we call the set of "Feasible Configurations", noting that $\mathbf{F}$ is properly a subset of $\mathbf{C}$ and $\mathbf{R}$ Cartesian product rather than of $\mathbf{C}$ itself.

With deepSPACE, we are not interested in only learning the feasible configurations, $\mathbf{F}$, but the entirety of the design space, including a wide range of configurations and requirements. This has several advantages. Your training set may have analogous solutions that generalize to new requirements. One trained model can be used for a wide range of uses, not only with a conceptual design loop but also with applications and projects.

C. Artificial Intelligence

AI or deep neural networks are an essential part of this framework. However, deepSPACE varies tremendously from the typical contemporary Large Language Models (LLM) models. One cannot simply use GPT-4 or otherwise adapt

them for aerospace design. This is due to obvious constraints. LLMs are notoriously bad at basic mathematics[12], they are trained as general tools for all tasks with little training towards aerospace, and most importantly, format restrictions mean that the underlying generative model must be able to work with numbers rather than words or characters.

We have decided not to present a specific AI model for this paper. Multiple types of Generative AI models can be used with this deepSPACE methodology. One could adapt Generative Adversarial Networks [13], Variational Auto Encoders [14], Graph Generators [15], or even a Generative Bayesian Network [16]. Each, of course, requires slight adaptations to the task at hand. The key thread is the ability to sample from a distribution given conditional requirements. This distribution, as we find in the next section, represents the style of the system. We envision the underlying AI model evolving over the following years as advances throughout the AI community trickle into the overall stack.

With neural networks of any kind, we should always expect bias/variance tradeoff [17]. This phenomenon means that there is a fundamental tradeoff in generalizability vs. accuracy. This has a follow-on effect on these generative models. Generally, a model that can generate a wide variety of configurations has more errors than the required ones. Training should continue, or model sizes should be updated until the model converges to within an acceptable error level. This is not an issue for conceptual design, as designers expect errors with conceptual-level analyses.

D. Design Space Slicing

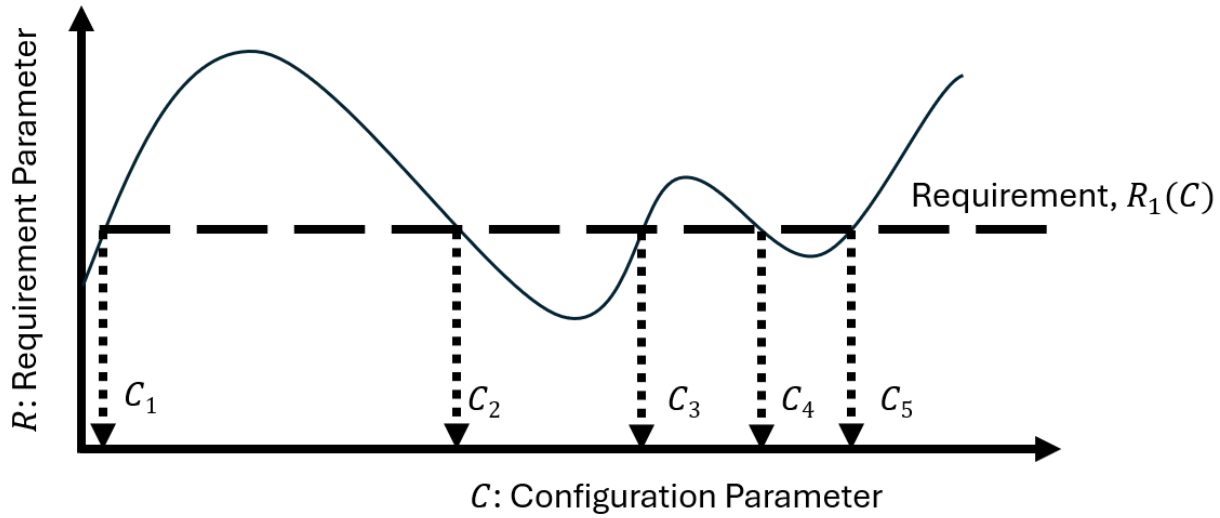

Fig. 1 Configuration vs. Requirement Parameters

The key breakthrough of deepSPACE in the field of Multi-Disciplinary Optimization is the ability to understand the vastness of the design space under a new paradigm. Typically, designers work in the space of parameters (C) and performance (R) as in Figure 1. They identify which parameters meet the target requirements, as in Figure 1 where the dashed line intersects the functional performance. Instead of sampling configurations to find performance, we want to find which configurations meet the required performance. To solve, we need extra information, we need a mapping that helps us identify the crossing points.

To get this extra information, we introduce a new variable. This is the Design Style variable (Z) as shown in Figure 2. This variable allows for a unique mapping of Configurations Parameter (C), Requirements (R), and Design Style (Z). The designer can now work with style and performance requirements to get to a design. To see how we find the crossing points, imagine as in Figure 3 we (z) and a requirement (r) when combined, this leads to a unique functional value of (c). In 2-D space of (C) and (Z), this plots out a curve as shown by the dotted line in the C-Z plane.

Due to the probabilistic nature of ANNs when mapping from style and performance to the parameter space in Figure 2, what is truly being referenced is the probability of a given design parameter. Figure 4 visualizes this probabilistic nature. Setting the requirements (R) and the Design Style (Z) provides a probability distribution over the configuration space. The designers typically only interact with the parameters that maximize the likelihood. Thus, all parameters are possible, but the user is provided with the value that maximizes this distribution. This fundamentally allows for the switching between wildly different configurations.

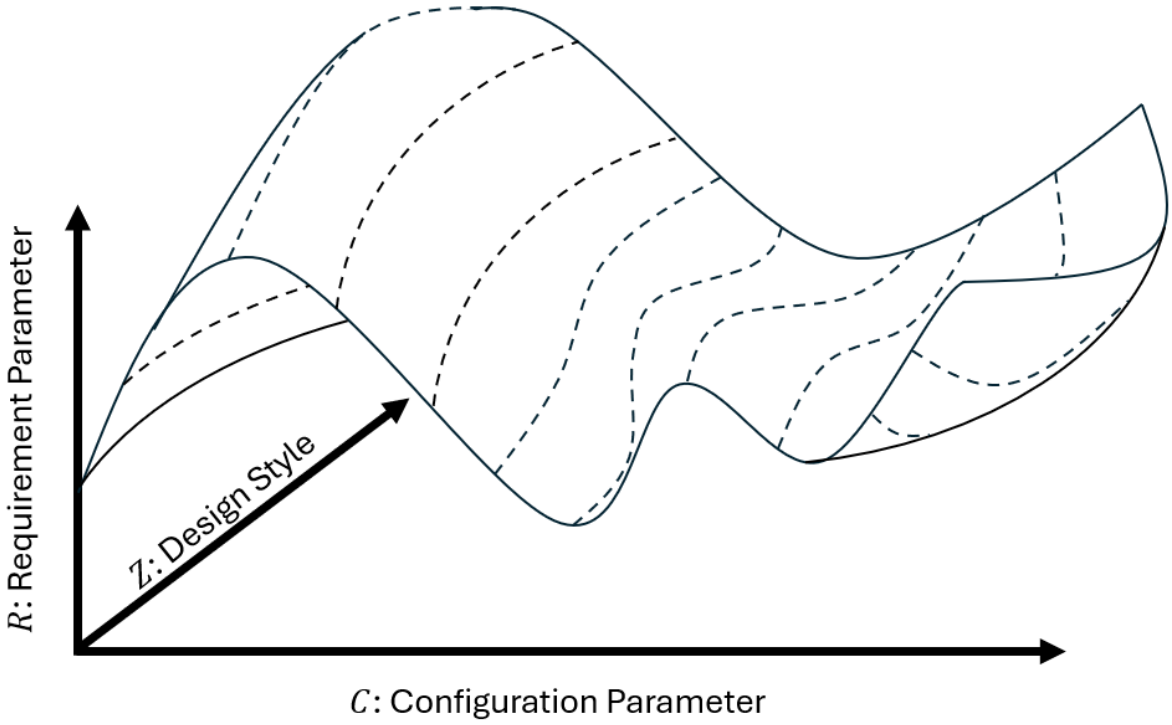

Fig. 2 Expansion of Style Dimension, Z

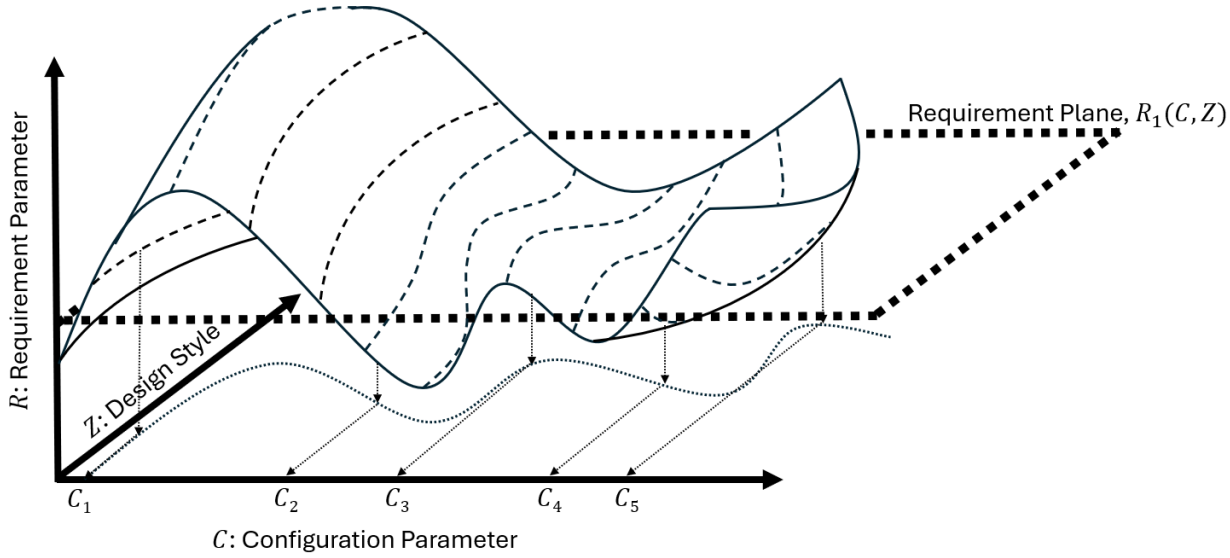

Fig. 3 Style Characteristics from Requirement Plane

The style space has several advantages over the parameter space. The style space is a one, or otherwise low dimensional, continuous representation, but critically NOT a relaxation, of the mixed integer space. Intuitively one can interpret the meaning of the styles visually, for example a designer can quickly discern the differences between the styles of a tilt-rotor and a compound rotor helicopter. Essentially, the style represents a key when combined with the requirements results in the parameters of the design or the configuration.

By using a style space representation of the design process, one can understand the behavior of the design space in

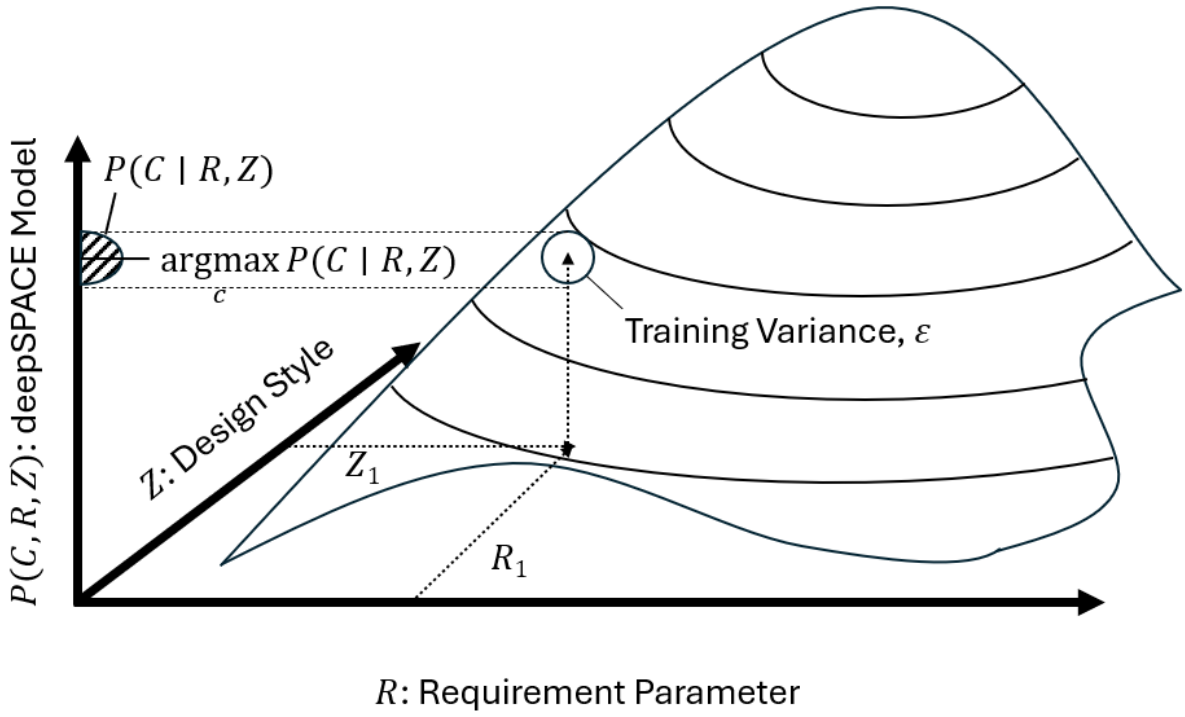

Fig. 4 Generative Model over Requirements and Styles

ways that were never possible before. We take inspiration from work done in molecular engineering[18]. If we take a slice of the design space at a particular requirements location, we can observe a range of design styles as in Figure 5. Depending on the domain, requirements, and the amount of searching allowed prior to inference, there may be many results or few. One feature of this is that along this line, the parameter typically changes smoothly locally until a jump in discrete configuration occurs.

This range of styles creates an opportunity to simplify the optimization process by following a slice of the requirements space through all possible styles, allowing one to perform a line search as shown in Figure 6. An intractable MINLP has been turned into a 1-D line search that can be accomplished with several methods. Depending on the resulting organization of the Design Style space and the chosen objective function, the functional shape as seen in Figure 6 may or may not be multimodal in shape. A gradient-based line search is unlikely to find the true optima for multimodal functions. With an unimodal or well-behaved functional shape for the design objective, we could use a gradient-based optimizer.

For this paper, we uniformly space queries of the style space and select the highest-performing values to select the optimal and semi-optimal candidate configurations. However, with no prior knowledge of the underlying structure, the proper technique would employ methods such as single-variable simulated annealing. We make no claims of guarantees of global optimality for all cases as it depends on the underlying generative model; however, we suggest this process efficiently identifies highly performant designs at the conceptual level compared to manual processes. We envision cases where designers may wish to perform local gradient-based optimization after conceptual design is performed with low-fidelity analyses. Such an optimization could be something like adjoint-based CFD for high-fidelity refinement.

When compared against existing methodologies used in aerospace design and beyond, it shows far more potential. We can see a comparison against the most commonly mentioned algorithms for these problems in Table 1. We can see that the instant restart, i.e. if objective functions change, one can immediately optimize again without retraining the model. This is because we can execute a limited number of queries to the MDO analysis to find the new objective of interest. Compared to traditional optimization processes, the limitation of this optimization process is that it requires upfront computational time to fully train and thus discover the novel configurations to ensure a near-global optimum. But it is on par with the number of existing MDO queries needed for optimization. We suggest that training is better used this time than in typical MDO cases. Especially since if one changes requirements, the process doesn't require retraining.

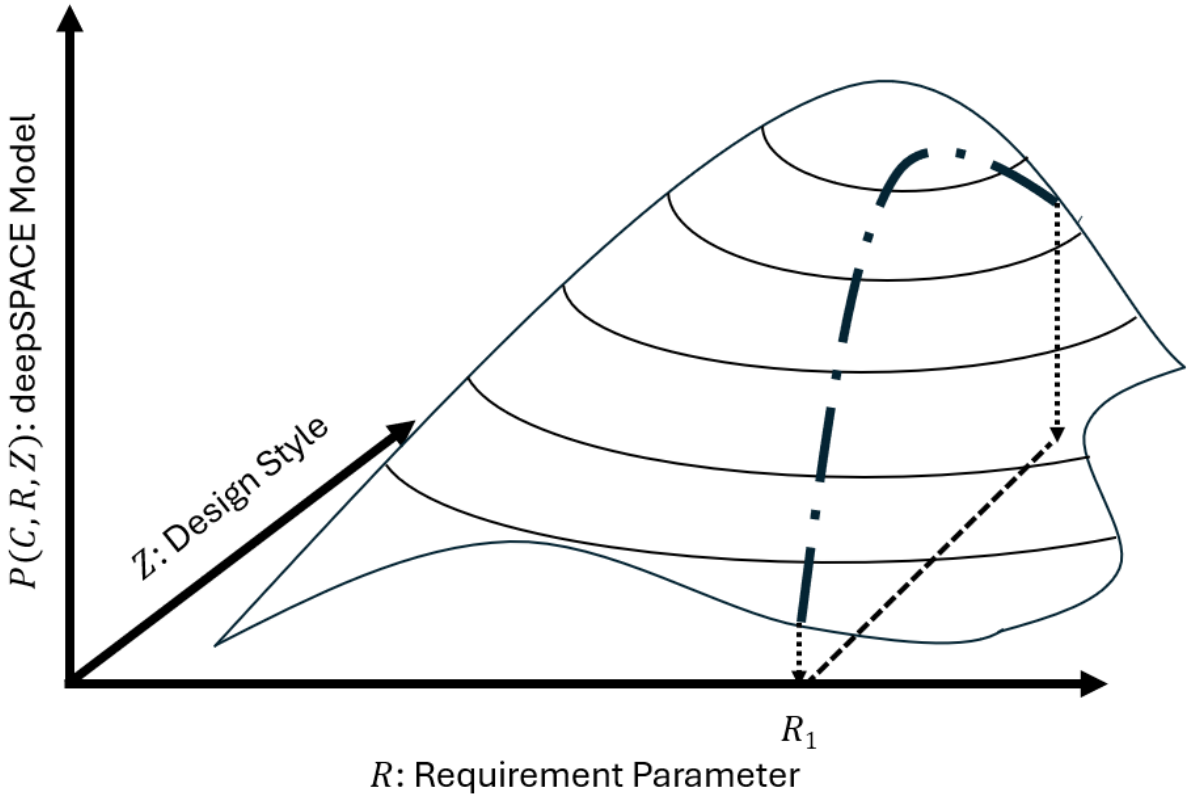

Fig. 5 Slice of Styles for Fixed Requirement

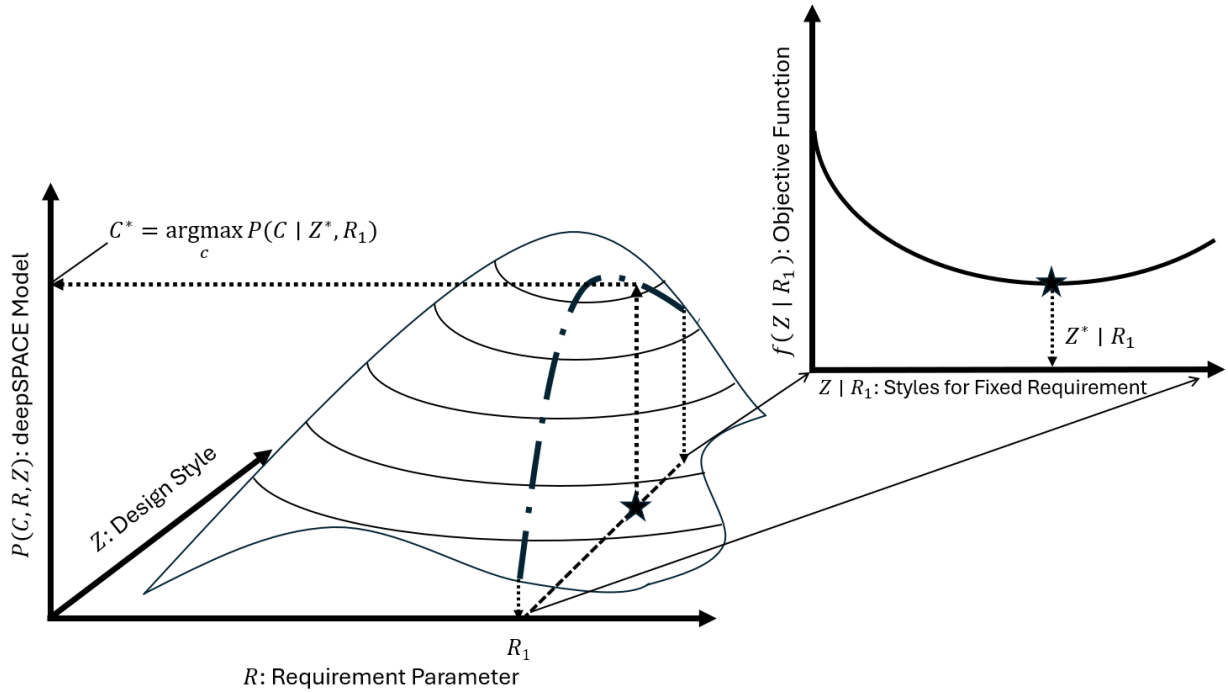

Fig. 6 Optimization over Styles for Fixed Requirement

For OEM design teams that perform constant design cycles, this means far less time to find an optimal solution.

E. Analysis

As with any MDO method, deepSPACE, in most cases, requires integrated analysis modules. Some examples are SUAVE [19], higher fidelity methods such as Finite Element or Computational Fluid Dynamics analysis, handbook analysis, traditional MDO tools, or really anything the user can find that can provide feedback. Unlike many MDO tools, deepSPACE was built to handle infeasible results such as NaNs and inf's. Typically, random samples can be used to build out synthetic data, much like when engineers do carpet plots, but they can be much more sparsely populated.

III. Results and Discussion

This section covers generated results using deepSPACE on several domains of interest, including structural design, automotive wheel design, aircraft design, and logistics supply design. Each is trained from totally different types of analyses, and the model is unique to that problem.

A. Structural Design

To demonstrate the ability of deepSPACE to rapidly synthesize structural designs, we first examine its performance on simple beam designs. For this, we assume a distributed load will be across a beam. The requirements are posed as a yield load and deflection at yield. The use cases of this are nebulous yet representative of reasonable requirements in multiple physical domains. For this case, we examine beams from a very wide range of requirements. This test case was selected for familiarity with aerospace engineers but exhibited highly non-linear effects.

1. Problem Formulation

The analysis is based on analytical Euler-Bernoulli beam bending equations for straight prismatic beams[20]. The boundary conditions are limited to cantilever and simply supported beams. The materials can be an isotropic "black carbon" fiber-reinforced polymer, steel, and aluminum. The cross sections are hollow ellipses, solid ellipses, I beams, hollow rectangles, and solid rectangles. In total, there are 30 combinatorial beam options. However, the selection of dimensions and the floating point dimensions themselves are also considered combinatorial options. These include width, length, height, and thickness as necessary.

The requirements are sampled between 10 Newtons and 1 giga-Newtons in yield load and 100 nano-meters and 10 Meters. This requirements space covers an enormous swath of the design space for beam designs. When coupled with the non-linearities of the beam analysis results, the configuration space is incredibly large as well. The model is initially trained with 76 samples for a short period of time on a laptop.

Table 1 Comparing Optimization Algorithms

	deepSPACE	**Gen. Alg.**	**Sim. Anneal.**	**Grad. Desc.**	**MINLP**
Mixed Integer	Yes	Yes	Yes	No	Yes
Black Box	Yes	Yes	Yes	Yes	No
Equality Constraints	Native	Penalty Function	Penalty Function	Native	Native
Inequality Constraints	Penalty Function	Penalty Function	Penalty Function	Native	Native
Global	Potentially	Potentially	Yes	Only Convex	Yes
Design Space Intuition	Yes	No	No	No	No
Instant Restart	Yes	No	No	No	No

2. Results

We can see the results of two sweeps in Figure 7. We pick a maximum displacement of 1 mm under yield load. Then, we sweep the style space, holding the yield load constant at 5000 lbs. Traversing across, we see some of the variants that meet the same requirements. Then, at the center where the solid rectangular steel beam is met, we sweep up and down the requirements space. Here, we can see how fixing the style produces visually similar and categorically the same beam but is rescaled to meet the adapting requirements. The designer can now quickly see the range of configurations, ranging from short stocky beams to longer solid beams, each of which is compliant with the requirements.

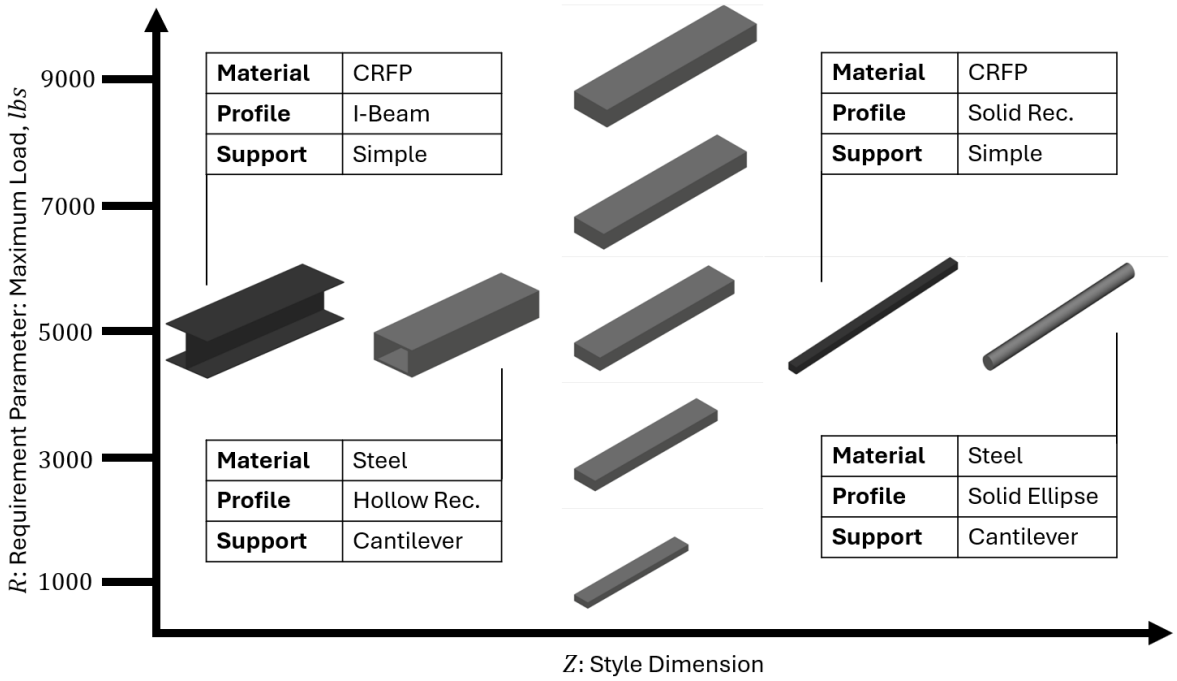

Fig. 7 Loading and Style Variation for Beam Design

B. Automotive Wheel Design

This example demonstrates deepSPACE's ability to perform aerodynamic optimization of automotive rims. These rims are subject to cost and weight requirements. The car weights and types are assumed to be in line with prior studies.

The aerodynamic properties are calculated from the work of Brandt et. al.[21]. This work combined both CFD and wind tunnel exploration to characterize the aerodynamic effects of parametric designs on automotive rims. The aerodynamics of these rims are provided by a regression of CFD results by changing key parameters of the rim geometry including the coverage area, rim cover distance, rim cover depth, rim cover angle, depth of center, drop angle, and if present a spoke window width. The regression is presented as a drag increment compared to a baseline smooth rim. Additional variables include material properties and the number of spokes.

The weights of the rims are calculated using sizing of the thickness of the rims and the coverage area. To size, the methods presented in Ugural [22] are used for hoop stress and vary with the rim size, material selection, and car weight. This sizing process produces an estimated weight for building our database.

Finally, the costs of manufacturing the rim are calculated and depend on the material type and the amount of cutting required. Rims with more cutout areas and more difficult machining properties have higher costs. The volume of the rim also affects the costs as more material is used.

A total of ten thousand samples are taken ranging from about $3 to over $10000, and weights from 3 kg to over 17 kg are used. These random samples are likely more data than necessary, but, given the simplicity of the problem, doesn't result in a major speed penalty.

1. Results

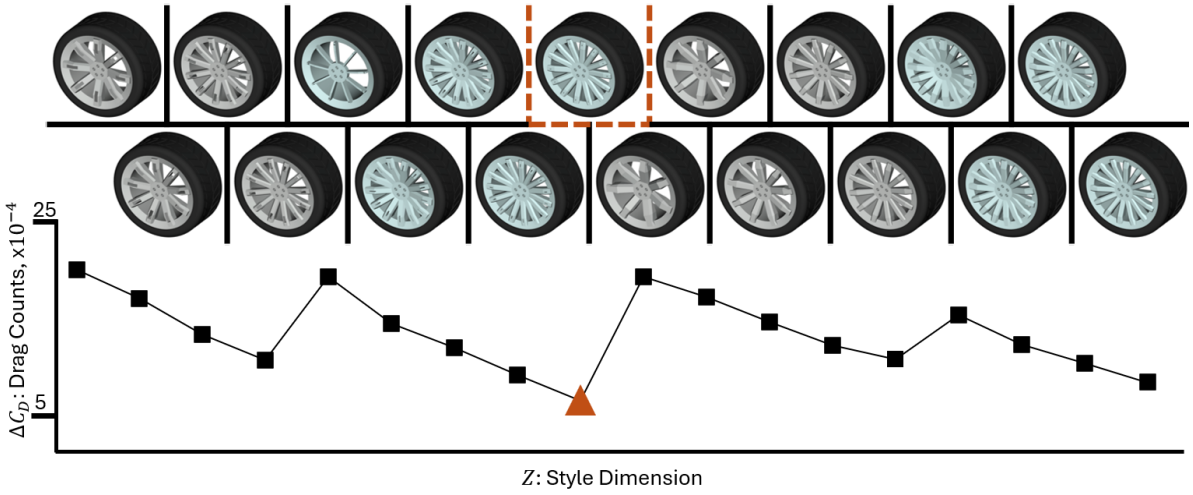

Fig. 8 Style Space for Automotive Wheel Design with Drag Analysis

To illustrate how users can interact with deepSPACE work, we will traverse a fixed set of requirements. We fix the requirements at 5kg in weight and $80 in manufacturing costs. The rims vary by not more than 5% compared to the target requirements. However, the rims are vastly different throughout the sweep seen in Figure 8. We see aluminum and magnesium wheels with various windows and spokes. We can plot the drag value compared to the baseline regressed wheel when sweeping the results. The highlighted rim is the most efficient given these requirements. It has the lowest drag due to the flat drop angle (recessed face) and extensive coverage (few gaps between spokes). We see that the design style produced smooth results locally, with breaks when large topology changes occurred.

C. Aircraft Design

Next, we demonstrate deepSPACE's ability to synthesize and optimize novel cargo transport aircraft. This domain has far more variability in viable designs than prismatic beams or car rims. The intention here is to produce results that are diverse rather than accurate when considering the bias/variance tradeoff. The results shown here demonstrate the tool's ability to credibly produce a design with little analysis rather than a commentary on the viability of a specific configuration.

The requirements are posed as the cruise range, Mach number, static margin, initial cruise C_L, initial cruise throttle setting, takeoff field length, payload volume, and payload weight. We begin training from five sample aircraft: a Boeing 737 class aircraft, a Boeing 777 class aircraft, an Embraer E-190 class aircraft, a 450-passenger blended wing body, and a Concorde class supersonic aircraft. We chose specifically conventional aircraft to highlight deepSPACEs ability to perform true synthesis of unconventional design configurations.

The components and their dimensional parameters, as shown in Table 2 can be used in any combination to design feasible vehicles. Main wings can be any complex shape and composed of any number of wing segments that are themselves vehicle components. The tail feathers are swept and tapered simple wings categorized as either Horizontal or Vertical tails for weight estimation. Energy networks are of fixed technology levels and sized solely from sea-level static thrust and turbofan bypass ratios. The engines have symmetry conditions enforced in SUAVE. The overall mass properties of the vehicle are themselves considered a component and fundamentally are the base class of the vehicle.

We configured SUAVE to perform a vehicle weight buildup using the SUAVE methodology, CG estimation, Vorlax-ported VLM analysis [23], and full mission simulation with reserve segments to estimate range, takeoff landing distance estimation using methods from SUAVE, industrial costs calculation, and static margin calculation at the beginning of cruise. This analysis, in total, takes about 4 seconds to execute on an older laptop, making use of parallel batches. A total of 204 feasible function evaluations are used to explore and train deepSPACE beyond the initial five samples of existing airliner designs.

Table 2 SUAVE Components

Component Name	Dimensional Parameters
Turbojet	Sea-Level-Static Thrust, X-, Y-, & Z-origin
Turbofan	Sea-Level-Static Thrust, Bypass Ratio, X-, Y-, & Z-origin
Main Wing	Projected Span, Root Chord Length, X-, Y-, & Z-origin
Main Wing Segment	% Span Location, Root Chord %, Twist, Dihedral, Quarter Chord Sweep
Horizontal Tail	Projected Span, Root Chord, X-origin, Y-origin, Z-origin
Vertical Tail	Projected Span, Root Chord, X-, Y-, & Z-origin
Fuselage	Length, Max Height, Max Width, Nose & Tail Fineness Ratios, X-origin, Y-origin, & Z-origin
Mass Properties	Maximum Takeoff Weight, Maximum Zero Fuel Weight

1. Results

In these results, we're looking for novel configurations that are unlike existing airliners. We set our requirements to 3000 statute miles in range plus a 45-minute IFR reserve, a cruise Mach Number of 0.88, a static margin of 20%, top of climb throttle of 0.9, an initial cruise C_L of 0.3, take of field length of 1000 meters, payload of 20 tons, and a cargo area of 200 cubic meters. Sampling throughout the Design Style provides a spectrum of results as shown in Figure 9. From conventional to unconventional. The first sample is an almost classic configuration but with a closely coupled tail. Upon further examination, it was found that it satisfied the static margin requirement, but the problem formulation didn't provide for a maneuvering constraint. Thus, this aircraft is stable but not maneuverable. Future iterations should include a maneuvering requirement.

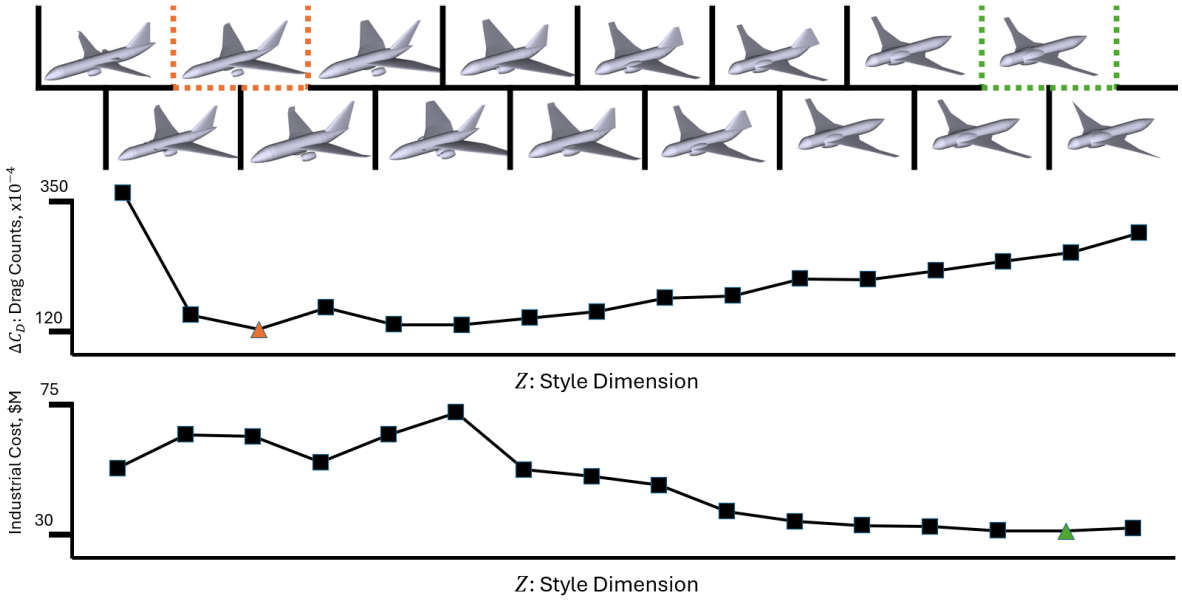

Fig. 9 Style Space for Aircraft Design with Drag and Cost Analysis

As we continue down, the next aircraft has a canard. A canard and wing shape are not present in the initial dataset. The configurations become progressively simpler as the fuselage shape morphs. The volumes stay constant, but they become stubbier before stretching out again. The final designs have simple, highly tapered wings and embedded engines.

Accompanying the images are two different objective functions, as shown at the bottom of Figure 9. One is the drag coefficient, and the other is industrial costs. Remember that the drag coefficient is normalized by the reference area of the vehicle, with each of these vehicles having different reference areas. We see that drag and industrial costs do not track with each other. Notice that the functional shape of drag and industrial costs are well-behaved. This is due to the inherent regularization of the Design Style domain. This means that similar designs are grouped together, ultimately

leading to smooth results. A designer would need to perform further studies in their conceptual design cycle to find the design that best fits the weighted objectives between drag, hence fuel burn, and costs industrial costs.

Beyond a fixed slice, we further explored what was possible in the design space. Trying many different styles and requirements. Eventually, we discovered something we thought was totally novel until we realized it was similar to an existing design [10]. This demonstrated to us that this tool is going far beyond mimicking prior art; there was nothing like this in the initial dataset, and no random sampling was taken to train this model. Therefore, we can confidently say that this did perform a true synthesis of a novel aircraft configuration.

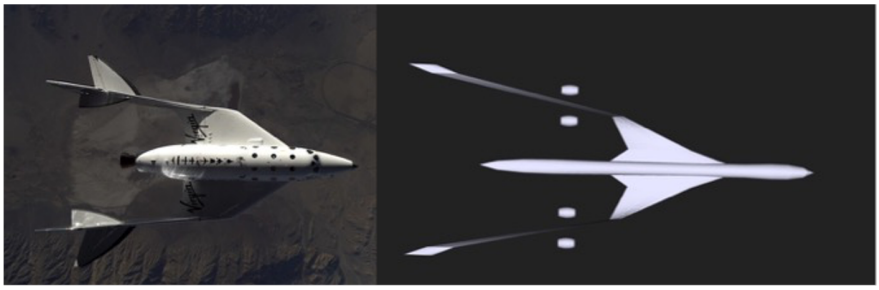

Fig. 10 Novel Concept Compared to Virgin Galactic Spaceship 2[24]

D. Logistics Design

As developed here, configuration design can be extended beyond cyberphysical system design. The same approach can be applied to aircraft, automotive, or structural design and operational requirements. As a demonstration of deepSPACE's extensibility, we consider a model logistics network design problem. The underlying network dynamics are simplified compared to those used in commercial airline network planning but maintain the highly combinatorial mixed-integer design space features that deepSPACE is intended to overcome.

1. Problem Formulation

The problem domain is specified as a hub-and-spoke model of the United States, including its hemispheric territories (i.e., Alaska, Hawaii, and Puerto Rico, but not Guam). The service operator has "bases" in New York, Los Angeles, Chicago, San Francisco, and Atlanta. The 50 largest US cities are reachable by air transport from each of these hubs (including bilateral air travel between the hubs). A further 1500 minor cities are reachable by ground transport from their nearest major city.

The design problem is determining a set of target cities/markets that may be efficiently serviced given a particular payload/shipping unit to be delivered and how many units to set to each target city. Each city has a local demand model which assigns a sigmoidal unit price based on the ratio of units to population, called the "demand factor", $n_i/P_i = d_i$:

$$U_i = \frac{U_{max}}{2}(1 - \tanh(10 * (d_i - 0.5))) \tag{1}$$

The revenue of the network is determined by summing $U_i \cdot n_i$ over every target city in the network. Costs are determined based on the volume of units trafficked and a per-unit cost for air transit and ground transit. Ground transit per-mile cost is determined by the number of twenty-foot equivalent units (TEUs) necessary to handle the volume. The final cost is a stepwise function of TEU mass, with \$3 per TEU-mile if TEU weight is under 3000 kg, \$8 per TEU-mile if between 3000 kg and 10000 kg, and \$13 per TEU-mile if exceeding 10000 kg.

Air transport costs, C_A, are dependent upon which aircraft deepSPACE specifies to be the uniform fleet vehicle, with volume, speed and distance-based fuel burn, f, for a nominal 737, 777, and 747 set of cargo aircraft detailed in Table 3. Cost per aircraft mile is taken as a quadratic function of TEU-volume weight:

$$C_A = 1E-7 \cdot f \cdot m_{TEU}^2 \tag{2}$$

Table 3 Transport Modes

Mode	Mass, t	Vol., m^3	Speed, kph	Fuel Burn, kg/km
737	23	141	850	3.3
777	53	653	950	6.9
747	139	963	930	9.9
TEU	11	33	100	N/A

2. Results

Here we fix the revenue to be \$1M and costs to be \$0.5M and sweep through five samples of the design style. The results are shown in Figure 11. What is notable about the results is the overwhelmingly different configurations presented to the user. Furthermore, the results are not typical CAD models; rather, they are displayed as a map. This methodology is not limited to traditional cyber-physical systems but can be used for designing and optimizing networks and other systems. The future potential of this is to combine the airliner design with the flight network to create an optimal airplane and network for current and future air passenger and cargo demands.

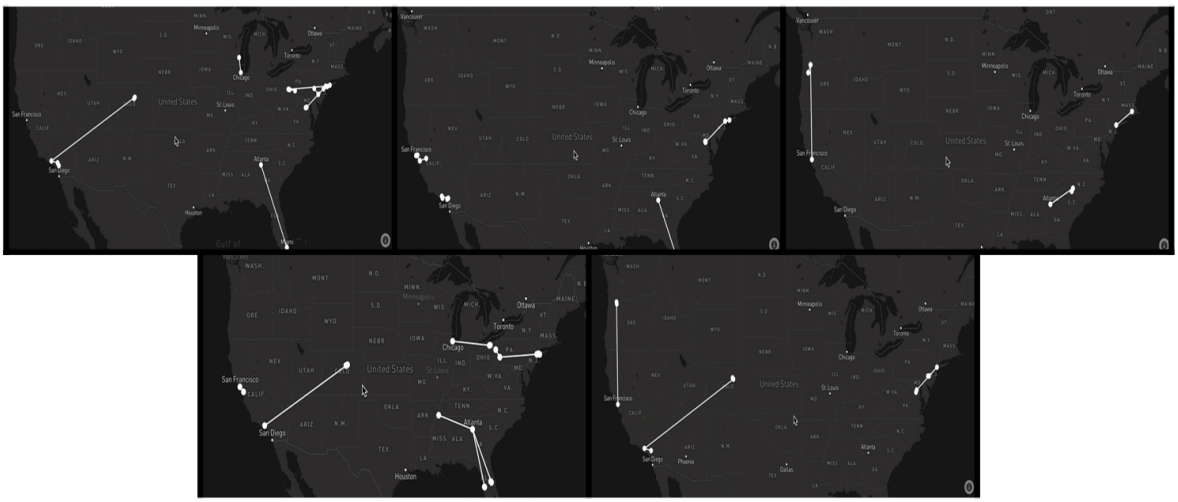

Fig. 11 Style Space for \$1M Revenue, \$0.5M Cost

IV. Conclusion

In this paper, we present the fundamentals of deepSPACE and use them to demonstrate design synthesis in several domains. Current state-of-the-art techniques for configuration design are typically a manual process, like brainstorming, guided by the intuition gained from designer experience. The deepSPACE methodology is built to empower the designer to rapidly gain intuition and optimize configurations, especially in scenarios with few existing solutions to gain inspiration from.

Conceptual design requires constant iteration between requirements, design, analysis, and customers. Closing cycles can be challenging in a reasonable time frame, leaving the designer to pick something and move on. Instead, by streamlining configuration design, we can give the designer confidence that they have selected the right thing.

Optimization can seem like a tantalizing solution to the configuration design problem. However, solving a black-box Mixed Integer Nonlinear Program is an NP-hard problem. Current solutions involve genetic algorithms and vast surrogate models to tractably solve. What designers lose with a pure optimization process for configuration design is understanding the design space they need. Optimizers can exploit weaknesses in the underlying analysis functions, leading to spurious results. When restarting the optimization process, when something changes, i.e., new requirements, these prior samples become a sunk cost in time and computational resources. Thus, most designers rely on carpet plots for large topology changes and gradient-based optimization for local changes if they do use them.

These issues motivated the creation of deepSPACE. By building a generative model for configuration design, we can decouple the requirements from the design style. This provides the user with an array of designs that they can visualize and objectively compare. We can slice through the design space and optimize the results. Instead of selecting only the optimal result, it exposes the ranked subordinate solutions. Designers can continue to iterate on requirements and explore, all without taking additional function evaluations.

To illustrate deepSPACE's ability to generalize to any domain of design, we present four different challenges. First, we examined a beam problem that displays how using the deepSPACE methodology, we can examine styles and requirements independently. Then, we perform aerodynamic optimization of automotive rims, which exhibits the optimization for a simple problem. Aircraft conceptual design and optimization are performed next with multiple objective functions. The aircraft conceptual design study showed its ability to synthesize totally novel configurations. Finally, we show how deepSPACE generalizes outside of typical engineering design to be used for operational planning problems.

This is just the beginning of this technology. We envision further refinements of this, including new AI models, new use cases, greater designer input into operations, and much more. We hope the engineering community, both at the industrial and research levels, is receptive to these ideas, adopts them, and refines them alongside us.

Acknowledgments

The authors would like to thank the countless designers throughout the aerospace industry and beyond for taking the time to sit down and chat with us. You have helped us understand how to improve the current state of the art in engineering design.

A portion of the material presented here is based upon work supported by the National Science Foundation under Grant No. 2333122.

References

[1] Sutton, R. I., and Hargadon, A., "Brainstorming groups in context: Effectiveness in a product design firm," *Administrative science quarterly*, 1996, pp. 685–718.

[2] Ilevbare, I. M., Probert, D., and Phaal, R., "A review of TRIZ, and its benefits and challenges in practice," *Technovation*, Vol. 33, No. 2-3, 2013, pp. 30–37.

[3] Pugh, S., "Concept selection: a method that works," *Proceedings of the International conference on Engineering Design*, 1981, pp. 497–506.

[4] Lee, J., and Leyffer, S., *Mixed integer nonlinear programming*, Vol. 154, Springer Science & Business Media, 2011.

[5] Burer, S., and Letchford, A. N., "Non-convex mixed-integer nonlinear programming: A survey," *Surveys in Operations Research and Management Science*, Vol. 17, No. 2, 2012, pp. 97–106.

[6] Mitchell, M., *An introduction to genetic algorithms*, MIT press, 1998.

[7] Smart, J. T., "Neural Heuristics for Mixed-Integer Configuration Optimization," Ph.D. thesis, Stanford University, 2023.

[8] Müller, J., Shoemaker, C. A., and Piché, R., "SO-MI: A surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems," *Computers & operations research*, Vol. 40, No. 5, 2013, pp. 1383–1400.

[9] Gage, P. J., Kroo, I., and Sobieski, I., "Variable-complexity genetic algorithm for topological design," *AIAA journal*, Vol. 33, No. 11, 1995, pp. 2212–2217.

[10] "ESDU 04008: Use of Carpet Plots to represent functions of two variables. — esdu.com," https://www.esdu.com/cgi-bin/ps.pl?sess=unlicensed_1100107121229jld&t=di&p=di_04008, ???? [Accessed 23-06-2024].

[11] Dekking, F. M., *A Modern Introduction to Probability and Statistics: Understanding why and how*, Springer Science & Business Media, 2005.

[12] Qin, C., Zhang, A., Zhang, Z., Chen, J., Yasunaga, M., and Yang, D., "Is chatgpt a general-purpose natural language processing task solver?" *arXiv preprint arXiv:2302.06476*, 2023.

[13] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y., "Generative adversarial nets," *Advances in neural information processing systems*, Vol. 27, 2014.

[14] Kingma, D. P., and Welling, M., "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[15] Faez, F., Ommi, Y., Baghshah, M. S., and Rabiee, H. R., "Deep graph generators: A survey," *IEEE Access*, Vol. 9, 2021, pp. 106675–106702.

[16] Botero, E. M., *Generative Bayesian networks for conceptual aircraft design*, Stanford University, 2019.

[17] Geman, S., Bienenstock, E., and Doursat, R., "Neural networks and the bias/variance dilemma," *Neural computation*, Vol. 4, No. 1, 1992, pp. 1–58.

[18] Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A., "Automatic chemical design using a data-driven continuous representation of molecules," *ACS central science*, Vol. 4, No. 2, 2018, pp. 268–276.

[19] Lukaczyk, T. W., Wendorff, A. D., Colonno, M., Economon, T. D., Alonso, J. J., Orra, T. H., and Ilario, C., "SUAVE: an open-source environment for multi-fidelity conceptual vehicle design," *16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2015, p. 3087.

[20] Timoshenko, S., *History of strength of materials: with a brief account of the history of theory of elasticity and theory of structures*, Courier Corporation, 1983.

[21] Brandt, A., Berg, H., Bolzon, M., and Josefsson, L., "The effects of wheel design on the aerodynamic drag of passenger vehicles," *SAE International Journal of Advances and Current Practices in Mobility*, Vol. 1, No. 2019-01-0662, 2019, pp. 1279–1299.

[22] Ugural, A. C., *Stresses in beams, plates, and shells*, CRC press, 2009.

[23] Botero, E. M., Clarke, M. A., Erhard, R. M., Smart, J. T., Alonso, J. J., and Blaufox, A., "Aerodynamic verification and validation of SUAVE," *AIAA SciTech 2022 Forum*, 2022, p. 1929.

[24] Galactic, V., "Photo from today: SpaceShipTwo moments before ignition of her hybrid rocket motor, as seen from WhiteKnightTwo." , January 2014. URL https://x.com/virgingalactic/status/421806784651603968.