Physics-Aware 3D Mesh Synthesis

Jianren Wang
Carnegie Mellon University
jianrenw@andrew.cmu.edu

Yihui He
Carnegie Mellon University
he2@andrew.cmu.edu

Abstract

This paper proposes a new task which emphasizes the importance of past-designs in 3D mesh synthesis. Instead of synthesizing novel meshes from scratch, we introduce a physics-aware 3D mesh synthesis algorithm, which consists of two modules: a 3D mesh synthesis module where we use a VAEGAN to encode 3D meshes into a latent variable and use the decoder to generate 3D meshes from the encoded representations; a scientific decision making module using reinforcement learning which alters the latent representation supervised by a provided physical constraint. The results show that our approach can modify a given mesh so that it satisfies external physical property constraints while maintaining high appearance similarity. More importantly, our method outperforms all baseline methods by a large margin.

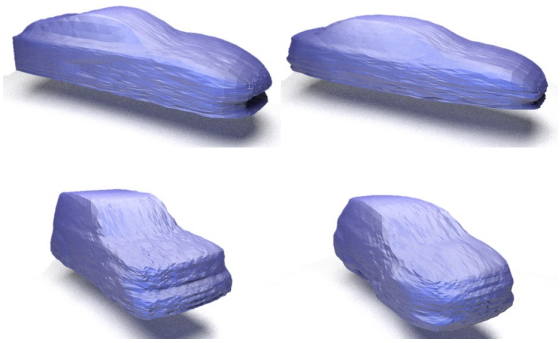

Figure 1. Corresponding results obtained by our learning model on automobile mesh data. All of the mesh models on the left side are the original data from the dataset, the mesh models on the right are generated by our learning algorithm which, assessed by OpenFOAM, have a smaller drag coefficient.

1. Introduction

3D shape synthesis is a long-standing and challenging problem in computer vision and computer graphics. In recent years, along with the exponentially growing industrial needs that range from virtual reality [52] to 3D printing, the capability for automated 3D prototyping becomes a critical technique that can potentially promote the development of several industries.

Recent researchers tackle to solve these problems through deep learning [28], since it has shown great success in image classification [25, 19], detection [20, 64] and segmentation [18]. With the introduction of large 3D datasets like ShapeNet [5, 62], many generative models can synthesize 3D objects that are both highly varied and realistic in voxels [59, 45], point cloud [12, 29, 33] and meshes [17, 31, 46, 53].

However, most existing methods focus on synthesising 3D shape from scratch (*i.e.*, propagate noise vectors through generative models). In a more realistic setting, many prototypes can be easily synthesised through modifying past designs to meet new constraints since they may encapsulate useful design information and are time-tested.

To emphasize this commonly used strategy in industry (synthesis through modification), we propose a new problem: meeting the external constraints with minimal modification of given shape priors. Specifically, the external constraints in this paper are physics properties, since these are commonly considered external constraints.

Previous researches show that deep learning enables end-to-end inference of object physical properties [4]. However, these methods can only understand simple physical properties such as velocity [58], stability [10], deformation [55] and are heavily constrained by specific domain. In contrast, computer-aided engineering tools based on Finite Element Method (FEMs) [43] and Computation Fluid Dynamics (CFDs) [21, 13] are designed to solve very complicated physics properties, which are highly accurate and are more adaptable to different domains. Using computational analysis software available (*e.g.*, OpenFOAM, ANSYS, etc.) we can analyze the physical properties (*e.g.*, drag coefficient, thermal conductivity, strain-stress relationship, etc.) given the shapes of the object and some predefined properties (*i.e.*, material, temperature, force application etc). However, these methods are always not differentiable. To this end, we propose an automatic physics-aware 3D mesh modification

method that utilizes both the accuracy of CAD tools and the efficiency of deep learning.

We start with finding a good encoder and decoder given the fact that meshes are very high-dimension data and modifying vertex by vertex is not feasible. We adopted a variational autoencoder with a generative adversarial network (known as VAEGAN [27]) to encode meshes into embedding vectors and decode embedding vectors into high realistic meshes. We then modify the embedding vectors using trust region policy optimization (TRPO) [40] so that the decoded mesh can meet external constraints (*i.e.*, drag coefficient) while maintaining high appearance similarity. The agent is supervised by high accuracy CAD tools. During inference, we can simply do forward propagation without any online tuning to achieve a very fast speed.

Our approach offers two unique advantages. First, by modifying meshes at feature level, we can largely alleviate the computational burdens and utilize more semantic information. For example, the embedding vector may encode some structures of some designs that have certain natures, which can be used to modify the mesh in a more realistic way. Second, by introducing reinforcement learning, our model can be trained end-to-end with CAD tools, which are always not differentiable. Thus, our method can achieve a good speed accuracy balance.

In summarize, this paper makes three contribution: we propose a new task that stress the synthesis by modification strategy; we propose an efficient physics-aware 3d mesh modification method; we demonstrate its effectiveness on ShapeNet [47] and our method outperforms baseline methods by a large margin.

2. Related work

Data-Driven 3D Shape Synthesis Before the deep learning era, some prior works on data driven 3D shape synthesis have proposed composing new objects from libraries of parts [14, 50]. These methods typically generate realistic shapes, but have a limited model expressiveness. More recent deep network approaches [6, 15] can generate higher variability of shape by learning deep generative 3D shape models on large datasets of synthetic CAD data (ex. ShapeNet [62]), but generally do not optimize for structural realism of generated shapes explicitly. Some other works continuously interpolating between shapes, such that the intermediate shapes maintain their semantic class [63]. The most commonly used structures are Variational autoencoders (VAE) [24] and generative adversarial networks (GAN) [16]. The former focuses on reconstructing overall shapes, while the latter focuses on generating synthetic shapes that can not be distinguished from real ones by a classifier. Both methods focus on appearance properties. In a more realistic setting, 3D shape synthesis should emphasize on both intuitive creativity and calculated scientific decision-making. The former stress the appearance properties (e.g. shapes, textures, etc) while the latter stress the physics properties (e.g. materials, stability, etc). Our focus is injecting new capabilities to these models of reasoning about physics properties. In this paper, we synthesis 3D shapes using meshes, since the points of each patch are stored within the same vicinity unlike point clouds [37, 38, 2] and are naturally adapted to CAD/CAE tools. Additionally meshes preserve the advantage of storage efficiency unlike 3D voxel [60, 32, 39, 54, 62].

Physics Reasoning Physical reasoning has raised a lot of interest in recent years [34, 36, 65, 11]. There are also works tackle to analyze more complex physics using machine learning, like fluid simulation. Tompson et al. [44] accelerates the pressure projection in Euler fluid simulations using a neural network. Chu et al. [7] use a Siamese network to learn a distance metric between fine and coarse simulation to synthesize the details of smoke. Instead of solving the underlying Navier-Stokes equations, Physicsforests [26] predict of the behavior of fluid particles based on a model trained on a large set of simulations. DPI-Net [30] proposes to learn a particle-based simulator for complex control tasks in robotics, like manipulating fluids and deformable foam. Umetani [48] directly generates a time-averaged velocity field and pressure field from given object. Although we focus on fluid simulation, there are many other approaches attempt to reason about physics in structural analysis [56], materials failure and damage [1, 61], material constitutive properties [41] and mechanics enhancement [22]. Unlike previous works, our method focuses more on shape manipulation instead of predicting physics properties. In this sense, the most related work is Pteromys [49], which propose a human-in-the-loop interactive design method to optimize the flight trajectories of model airplanes.

Computer-Aided Design CAD is widely used in many engineering fields. Computational methods aid in understanding the mapping between shape and function. Dering *et al.* [9] proposed a method that predicts the functional quantities of digital design concepts. Oh *et al.* [35] combined generative methods with topology optimization in automobile wheel design. Specifically in CFD, one common software in use is Open source Field Operation And Manipulation (OpenFOAM) [21], which can be used to calculate the drag coefficient C_d of a mesh model.

3. Methods

In this section, we introduce our physics-aware 3D mesh synthesis method (Figure 2). It consists of two modules: 3D mesh synthesis module and scientific decision making mod-

ule. 3D mesh synthesis module adopts graph VAEGAN, which intends to find a good shape embedding and shape decoder. Scientific decision making module takes the shape embedding as input and output an action to manipulate the embedding so that the decoded mesh can meet external constraints while maintaining high appearance similarity. During inference, we can simply do forward propagation without any online tuning to achieve a very fast speed. We'll discuss each component in detail.

3.1. 3D Mesh Synthesis Module

Meshes are naturally adapted to CAD/CAE tools, and are thus suitable for physics reasoning. However, it is not feasible to directly manipulating meshes since they usually contains thousands of vertices. We thus propose 3D mesh synthesis module to encode meshes into embedding vectors and decode these vectors to get high fidelity meshes. We adopted VAEGAN with graph convolutional layers. The whole system contains an encoder (Enc), decoder (Dec) and discriminator (Dis). We first encode a mesh x into a latent representation z and decode the latent representation back to data space, respectively:

$$z \sim Enc(x) = q(z|x), \tilde{x} \sim Dec(z) = p(x|z) \quad (1)$$

The VAE regularizes the encoder by imposing a prior over the latent distribution $p(z)$. In our case, $z \sim N(0, I)$. The objectiveness of VAE is to minimize the reconstruction loss (L_{recon}) and a prior regularization term (L_{KL}):

$$L_{\text{recon}} = \|\tilde{x} - x\|_2 \quad (2)$$

$$L_{KL} = D_{KL}(q(z|x)\|p(z)) \quad (3)$$

, where D_{KL} represents the Kullback-Leibler divergence. For more details, please refer to [27].

A discriminator is used to ensure generated shape realistic. We use binary cross entrophy as the classification loss, and present the adversarial loss function as:

$$L_{\text{GAN}} = \log Dis(x) + \log((1 - Dis(\tilde{x})) \quad (4)$$

Formally, our overall loss function can be writen as:

$$L = L_{\text{recon}} + \alpha_1 L_{KL} + \alpha_2 L_{\text{GAN}} \quad (5)$$

where α_1 and α_2 are weighting factors of KL divergence loss and adversarial loss.

The internal details of the encoder (Enc), decoder (Dec) and discriminator (Dis) are largely influenced by the choice of the 3D shape representation. To preserve the connection information in the mesh data, our method is primarily composed of the dynamic filtering graph convolutional layers proposed in FeaStNet [51]. The input to the layer is a feature vector field on the mesh vertices, where each vertex i is attached a feature f_i^{in}. The output is also a vector field f_i^{out}, possibly of a different dimension, computed as:

$$f_i^{out} = b + \sum_{m=1}^{M} \frac{1}{|N_i|} \sum_{j \in N_i} q_m(f_i^{in}, f_j^{in}) W_m f_j^{in} \quad (6)$$

, where N_i denoting a patch that contains the vertex i, $q_m(f_i^{in}, f_j^{in}) \propto \exp\left(u_m^T(f_i^{in}, f_j^{in}) + c_m\right)$ are the positive edge weights in that patch and normalized to sum to one over m. W_m, u_m, c_m and b are trainable weights, while the number of weight matrices M is a tunable hyperparameter [51]. Please refer to Figure 2 and [51] for more details.

3.2. Scientific Decision Making Module

Scientific decision making module takes the shape embedding as input and output an action to manipulate the embedding so that the decoded mesh meets the external constraints with minimal changes. In our case, we focuses on drag coefficients for automobile models. The drag coefficient is a scalar representing the drag or resistance of an object in a fluid environment, which is an important design variable for automobiles. Intuitively, the smaller the drag coefficient the more streamline the object is. The drag coefficient is defined as

$$C_d = \frac{2F_d}{\rho u^2 A} \quad (7)$$

where F_d is the drag force (in the direction of flow velocity), ρ is the mass density of the fluid, u is the flow speed of the object relative to the fluid and A is the reference area. We use OpenFOAM (F_{op}) — an open source CFD software to obtain the drag coefficient (C_d) from given mesh (x): $C_d = F_{op}(x)$. After assessing the drag coefficient of the 3D generated mesh, we introduce a one step reinforcement on mesh embedding space to reduce the difference in drag coefficient between the target and the generated model.

Formally, we denote the policy of scientific decision making agent as π, the state of agent as s, and the action of agent as $a \sim \pi(s)$. The state includes both mesh embedding $z \sim Enc(x)$ and target drag coefficient C_d^t:

$$s = [z, C_d^t] \quad (8)$$

Specifically, we concatenate z with C_d^t in the vector space.

The policy is trained to minimize the difference between C_d of synthesized mesh and C_d^t as well as the difference between modified mesh and original mesh:

$$\pi = \arg\min_{a \sim \pi(s)} \|F_{op}(Dec(a+z)) - C_d^t\|_2 + \lambda \|Dec(a+z) - x\|_2 \quad (9)$$

, where λ is a weighting factor to balance the aforementioned two differences.

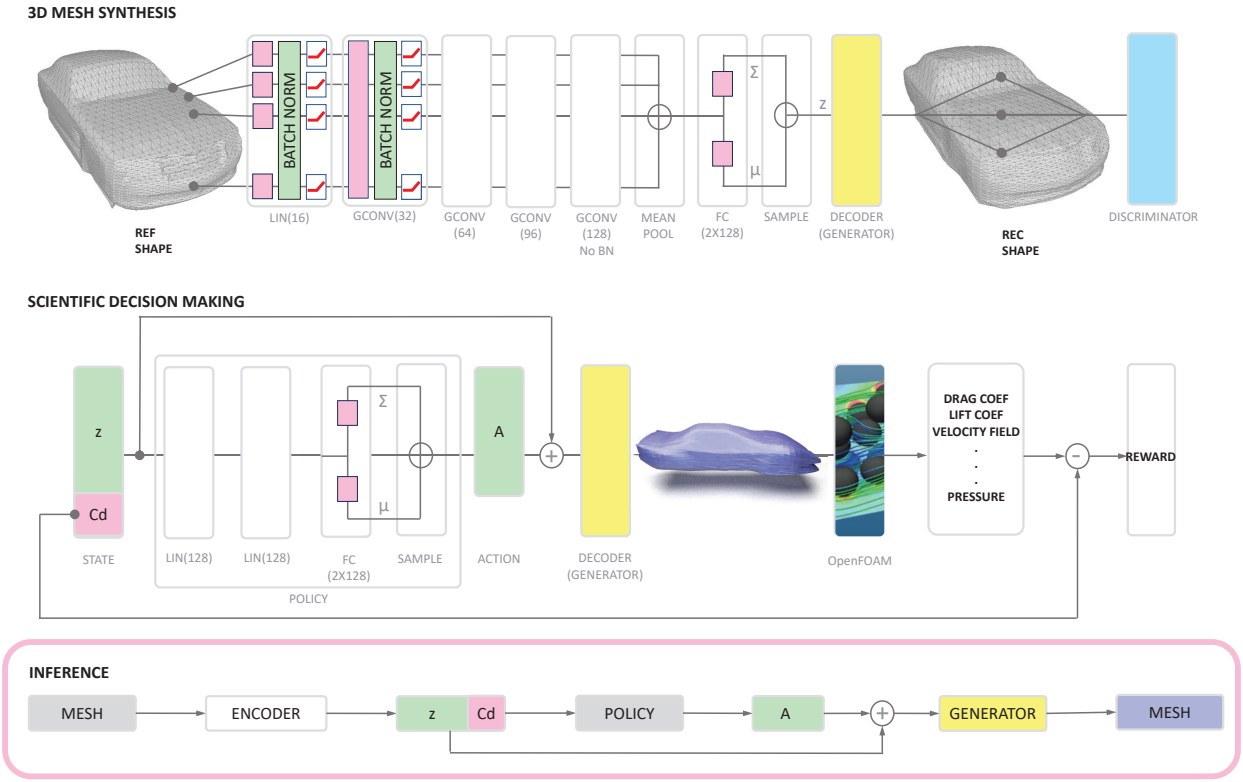

Figure 2. At the top we have our 3D mesh synthesis module which intends to encode and decode a high-dimensional mesh, in the middle we presents our scientific decision making module which modifies mesh embedding z to satisfy the external constraints with minimal modification of given shape priors. In the bottom we show the inference of our method.

Please note the action a has the same dimension as the mesh embedding z, and thus can be added to z to generate a new embedding. During training, the reward is set to the negative of the objectiveness, which is:

$$r = -(\|F_{op}(Dec(a+z)) - C_d^t\|_2 + \lambda\|Dec(a+z) - x\|_2) \tag{10}$$

We use trust region policy optimization (TRPO) to optimize our policy $\pi(\mathbf{a}|\mathbf{s})$.

3.3. Inference

During inference, we adopt the encoder Enc, decoder Dec from mesh synthesis module and the policy π from scientific decision making module. Given original mesh x and target drag coefficient C_d^t, the physics-aware mesh x' can be synthesized using:

$$x' = Dec(\pi([Env(x), C_d^t]) + Enc(x)) \tag{11}$$

Without any online tuning, our proposed method contains only forward propagation and thus achieves a very fast inference speed.

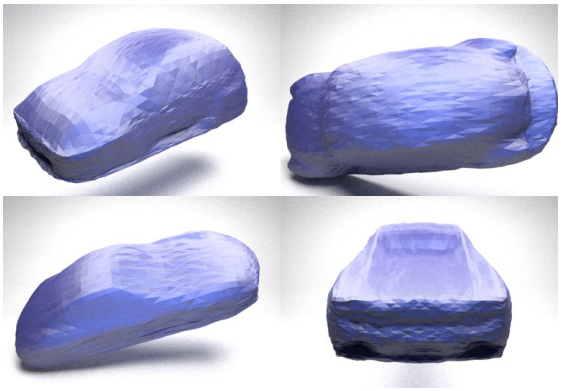

Figure 3. Random automobile shapes generated by the 3D mesh synthesis module. We explicitly relaxed the Gaussian prior on the latent variables during training to obtain a variety of automobile shapes. With the usage of GAN we can ensure that the generated automobile models are realistic even with this relaxation.

3.4. Technical Details

Dataset We use automobile shape data as proposed in [47] for training and validation, which consists of 1241 different automobile meshes spanning all automobile categories. We manually modify the shape to remove the side

mirrors, spoilers and tires. We divided the data into a training set which has 868 samples and a validation set which has 373 samples.

Mesh Synthesis Network The encoder Enc includes a linear layer (16), graph convolutional layers (32, 64, 96 and 128), and a mean pooling layer. The number denotes the channel number of each layer. Each graph convolutional layer is followed by a batch norm layer, a ReLU layer with the exception of the last one, which is not followed by a batch norm layer. The decoder and discriminator shares a common architecture by inversing the encoder. The last linear layer of the decoder has 3 channels indicate the coordinates of each point while the last linear layer of the discriminator has only two channels indicate the binary classification.

Decision Making Network To estimate the drag coefficient, we first simulated 10 seconds of air flow, due to the instabilization issue with the initial air flows we take the results for the last 4 seconds and used the mean. We simulate the fluid flow using OpenFoam Navier-Stokes solver with the $k-\epsilon$ turbulence model [57] and SUPG stabilization [42]. The simulation runs on tetrahedral mesh that conforms to the boundary. We made the mesh around the surface and the rear of the car to be finer in order to resolve the boundary layer and separated vortices resulting from non-slip boundary condition. To obtain a more realistic result we set the car to drive in an air speed of 72 km/h, which has $Re = 5 \times 10^6$.

The input layer of TRPO has 256 channels which is the same as state dimension. We duplicate the target drag coefficient 128 times before concatenating with the mesh embedding (128-dim), which forms a 256-dim state vector. We want the decision making agent to focus more on the target drag coefficient.

Training We trained our mesh synthesis network and decision making network separately. For mesh synthesis, we trained the model directly on the $3 \times N$ input meshes obtained from the automobile shape dataset as described above. The data is augmented by adding normally distributed noise to the vertices positions as well as a global translation and scaling planar. In each iteration of the training, we first get a sample z from the latent distribution. Then we update the encoder Enc, the decoder Dec, and the discriminator Dis sequentially. We use the ADAM optimizer [23] with the learning rate of Enc set to 10^{-4}, Dec set to 10^{-4} and Dis to 0.5^{-4}. We train the synthesis module for 5×10^5 iterations with a batch size of 8.

For the decision making network we used TRPO. We first sampled 16 latent variables from the variational distribution generated by the encoder, and then treated them as the whole dataset. For each of these small datasets, we trained the network for 100 epochs with a learning rate of 10^{-3}.

4. Experiments

In this section, we demonstrate results for both 3D mesh synthesis and scientific decision making. We trained our algorithm on the automobile shape data. The Physics-Aware VAEGAN is trained on data from [47], which consists of 1241 different automobile meshes spanning all automobile categories with the exception of cargo trucks. We performed a thorough comparison for the 3D mesh synthesis quality of our method with a variety of benchmarks, and we evaluated the performance based on root mean squared error (RMSE), estimated physics parameters, and volumetric error.

4.1. 3D Mesh Synthesis Results

Using VAEGAN combined with graph convolutional layers, we can generate models with high variance, as shown in Figure 3. We can observe that our algorithm preserves both high-frequency information (*i.e.*, the bump on the hood) and low-frequency information (*i.e.*, car ratio) in the mesh data. Our algorithm also generalizes this high-quality 3D mesh synthesize to a variety of automobile shapes.

In terms of reconstruction, our method achieves very low reconstruction losses. We compare our method with VAE (FC), which concatenates all vertices together as a large vector and VAE (Graph Conv), which doesn't use discriminator. The quantitative results are presented in Table 1, where RMSE represents the average displacement for each of the vertices, and the volumetric error represents the difference in volume size. As we can observe, our approach produces the best outcome in both RMSE and volumetric error.

Figure 4 shows synthesized meshes as the result of linear interpolation in the latent space. The source (left-most shape) and target (right-most shape) 3D shapes are passed through the encoder to obtain the corresponding latent representation, then we did a linear interpolation between the source and target latent representation, finally we passed these latent variables through the decoder to produce a highly non-linear interpolation in the 3D shape space $\mathbb{R}^3$. Many encoding methods would produce latent variables that do not correspond to any meaning full data in the presence of small variations. However, we demonstrate our approach can produce meaningful latent representations even when interpolation between two latent variables (Figure 4). By using VAEGAN we can more efficiently utilize the latent variables.

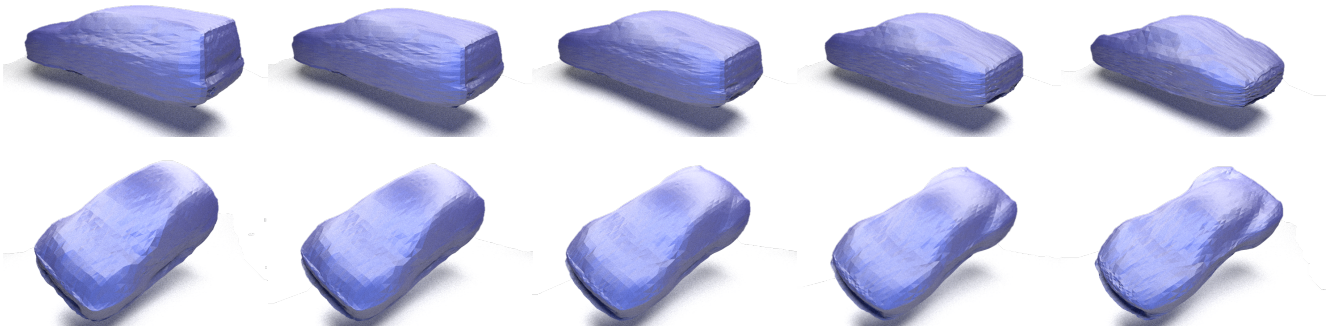

Figure 4. **Latent space interpolation.** Interpolation between two automobile mesh models (left- and right-most mesh models) obtained using a linear interpolation in the latent space.

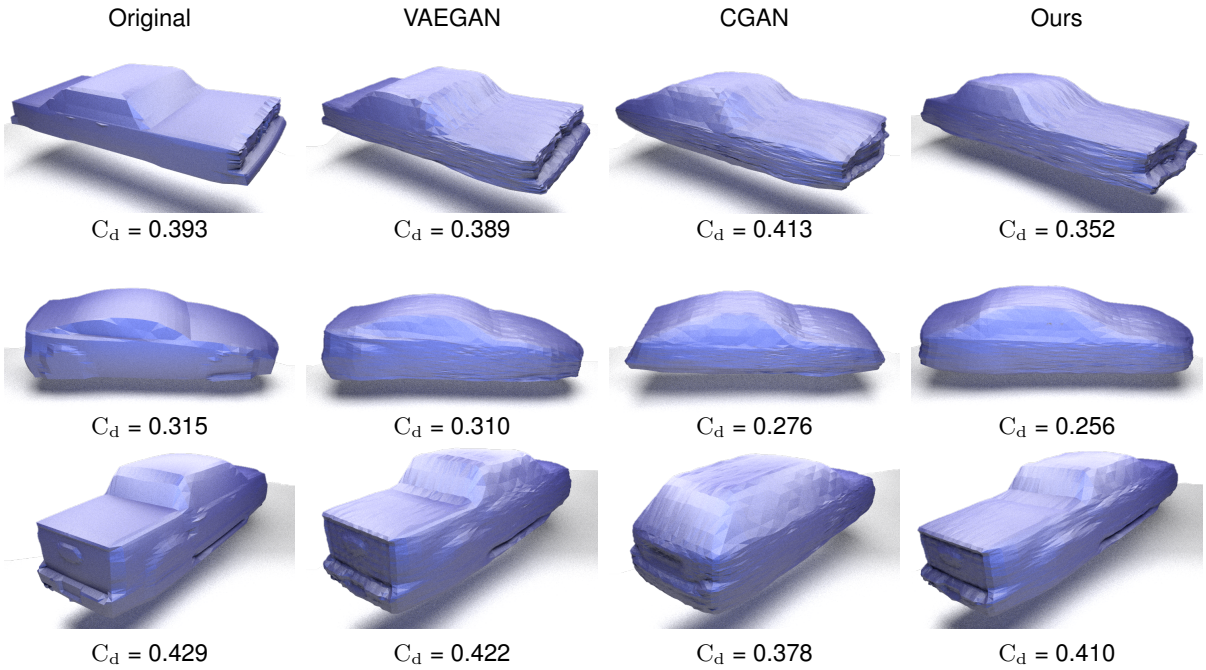

Figure 5. This figure compares the drag coefficient of the synthesized 3D shape between using VAE, Conditional GAN (CGAN), Ours and the original mesh data. For the three rows, the target drag coefficients are set to 0.35, 0.25 and 0.40, respectively. The goal for CGAN and Ours is to reach the target drag coefficient by exploring latent space. Note that the generation for VAEGAN and the original data does not take in the drag coefficient as a constrain.

Table 1. 3D Synthesis Error

Method	RMSE(m)	Volumetric err. [%]
VAE (FC)	11.1×10^{-2}	9.35
VAE (Graph Conv)	3.6×10^{-2}	4.03
Ours	3.3×10^{-2}	3.39

4.2. Scientific Decision Making Results

We compare our proposed scientific decision-making module with the baseline conditional GAN method (CGAN) [8]. Our module is trained and tested in the way, as mentioned in Section 3. For CGAN, during training, we randomly choose a mesh whose drag coefficient is within a threshold of the target drag coefficient as positive label. The discriminative model is trained to distinguish the synthesis mesh from positive mesh. We randomly sample from $[C_d(original) - 1.0, C_d(original) + 1.0]$ as target drag coefficient of corresponding mesh.

Evaluation of our success consists of two parts. One is the ability to satisfy the physical constraint in the form of drag coefficient, the other is the intuitively correct design

Table 2. Scientific Decision Making Error

Method	C_d Error	RMSE(m)
Conditional GAN [8]	2.21×10^{-1}	4.46×10^{-1}
Ours	1.46×10^{-1}	1.87×10^{-1}

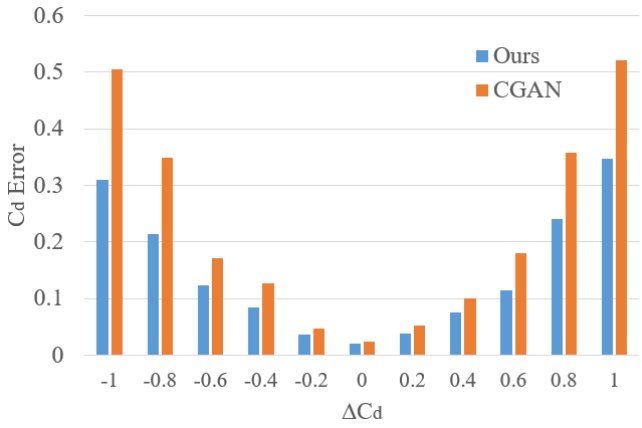

Figure 6. This figure shows the error distribution where the x-axis represents the difference between the target and original 3D shape drag coefficient and the y-axis represents the difference between the drag coefficient for the synthesized mesh and target drag coefficient.

based on appearance. Our final goal is to use the latent variable representation of the original automobile mesh model to generate an automobile mesh model that is both close to the original model while also satisfying the physical constraints.

During inference instead of sampling, we take the mean of the output action probability density function. By adding the action to the latent variable, we obtain a new mesh embedding, which is later passed through the decoder for synthesizing the target mesh. We calculate the drag coefficient of the target mesh using the same setup as mentioned in Section 3. We report the absolute error between the drag coefficient of the target mesh and target drag coefficient, which is shown in Figure 6. We observe that when the target drag coefficient is farther away from the original drag coefficient, the error is also larger. This means our model can modify the shape more accurately when the δC_d is small, which is very easy to understand. We also observe consistent improvement above the baseline method under all cases.

Table 2 depicts when compared with CGAN, our drag coefficient reconstruction error is 50% less, this shows that our policy network can more accurately find the direction of change in the latent space to satisfy the constraints in fluid dynamics. On the other hand, we also report the RMSE distance between target mesh and original mesh, which further prove that our method can achieve a higher C_d accuracy with relatively less modification. Additionally, compared to CGAN our approach produces more realistic 3D shapes.

Figure 5 shows when we keep the type of automobile in the same category using our approach, CGAN will attempt to generate a 3D shape that belongs to a different automobile type for lowering the drag coefficient. We can clearly see in the bottom row when we are optimizing for pickup trucks the result of CGAN is an SUV. A limited amount of data for pickup trucks in the dataset is the main reason that causes difficulties for CGAN to capture edge cases. However, our model learns general knowledge about modifying shapes to fit target drag coefficients and thus alleviate the problem of overfitting to training dataset and can better generalize to novel shapes.

As we can see from Figure 7, our approach has the ability to change the drag coefficient based on the provided target drag coefficient. The scientific decision making module learns to both manipulate local and global features of the 3D shape in altering its fluid dynamic properties. When the target coefficient makes our scientific decision making module to increase the drag coefficient, we noticed a widening in the car front, a steepening in the front surface and also increasing the angle of elevation for the tail. When the goal is to decrease the drag coefficient, the synthesized 3D shape tends to be more streamlined, decreasing the surface of the front face and lower the angle of elevation for the tail. All of these changes align with common physical intuition.

5. What does the scientific decision making module learn?

It is interesting to understand why the scientific decision making module can change the embedding vector to fit target physical parameters. In this section, we provide a more detailed analysis on what "physical law" this module learns through analysis of the pressure field before and after physics parameters fitting using Paraview [3].

In Figure 8, it shows the relative pressure of the pressure field of two meshes. We set the target drag coefficient to 0.35, and the original 3D shape (the top one) has a drag coefficient of 0.418. The synthesized mesh has a drag coefficient of 0.348, which is closer to the target drag coefficient. By analyzing the pressure field of the original and the improved meshes, we could see the policy network has learned to affect the pressure field and reach the target drag coefficient through controlling the shape of the mesh. By making the back of the car more tilted, the recirculation zone of the improved mesh is smaller than the original one which helps in reducing the horizontal force applied on the vehicle. Also, by smoothing the outline of the mesh, we could see the pressure field of the improved 3D shape becomes smoother. The boundary layer of the shape is also becoming thinner to reduce the force on the mesh. Reducing the front area of the hood makes the pressure distributed more evenly on the front area, thus helps with decreasing the total horizontal force on the mesh. In figure 9, the angles of

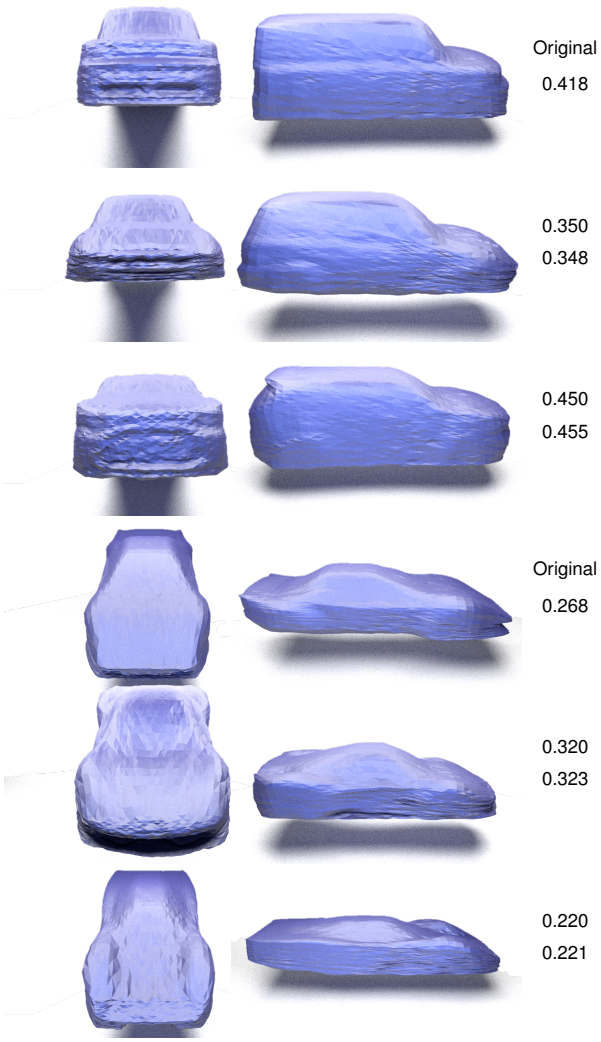

Figure 7. Our model has the ability to reach the target drag coefficient according to the given drag coefficient. For the right-most column, we have the target drag coefficient on the top and the drag coefficient of the actual drag coefficient of the synthesized model in the bottom. Also, the front of the car became more streamlined to achieve smaller drag coefficient. While to have a larger drag coefficient, the synthesized mesh developed thicker front. These alternations align with our physical intuition.

attack for the front and back of the original mesh are altered to make the pressure difference of the mesh smaller. In conclusion, our policy network has obtained the sense of pressure field around the synthesized mesh and utilized it to change the shape in reaching the target drag coefficient.

6. Conclusion

This paper proposes a new task which emphasizes the importance of past-designs in 3D mesh synthesis. We also introduce a novel method that tackles to solve this problem. The algorithm consists of two modules: 3D mesh synthesis module where we use a VAEGAN to encode 3D meshes into a mesh embedding and use the decoder to synthesize 3D meshes from the encoded representations; scientific decision making module using a policy gradient algorithm which alters the mesh embedding to meet the external constraints with minimal modification of given shape priors. Using a dataset of 3D automobile meshes, we validate our approach on two criteria: the ability to synthesize 3D shapes with the physical property constrained; the similarity of synthesized mesh with origin one in appearance. Experimentally we show that our approach can satisfy these two criteria. Additionally we show that our method outperform baseline methods in both 3D shape reconstruction and physical constraints satisfaction. One limitation is we only explored the drag coefficient for automobiles. But please notice our approach can easily adapt to similar physical properties (e.g. stability) for meshes in other categories. We leave to future work the task of extending to other properties on a variety of objects.

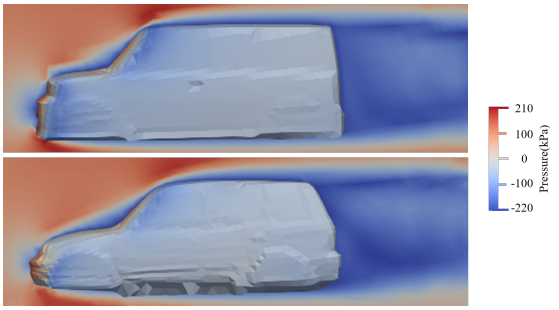

Figure 8. Pressure field before(top) and after(bottom) physics parameters fitting. Drag coefficient of the car on the top is 0.418, on the bottom is 0.348.

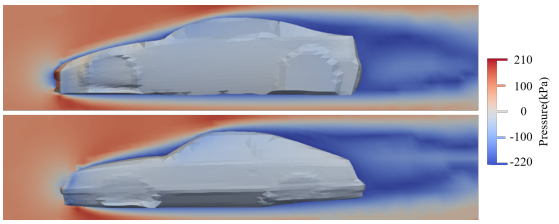

Figure 9. Pressure field before(top) and after(bottom) physics parameters fitting. Drag coefficient of the car on the top is 0.258, on the bottom is 0.203.

References

[1] M. Abendroth and M. Kuna. Determination of deformation and failure properties of ductile materials by means of the small punch test and neural networks. *computational materials Science*, 28(3-4):633–644, 2003. 2

[2] P. Achlioptas, O. Diamanti, I. Mitliagkas, and L. Guibas. Learning representations and generative models for 3d point clouds. *arXiv preprint arXiv:1707.02392*, 2017. 2

[3] J. Ahrens, B. Geveci, and C. Law. Paraview: An end-user tool for large data visualization. *The visualization handbook*, 2005. 7

[4] Z. Boukouvalas, D. C. Elton, P. W. Chung, and M. D. Fuge. Independent vector analysis for data fusion prior to molecular property prediction with machine learning. *arXiv preprint arXiv:1811.00628*, 2018. 1

[5] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1

[6] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 2

[7] M. Chu and N. Thuerey. Data-driven synthesis of smoke flows with cnn-based feature descriptors. *ACM Transactions on Graphics (TOG)*, 36(4):69, 2017. 2

[8] B. Dai, S. Fidler, R. Urtasun, and D. Lin. Towards diverse and natural image descriptions via a conditional gan. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2970–2979, 2017. 6, 7

[9] M. L. Dering and C. S. Tucker. A convolutional neural network model for predicting a product's function, given its form. *Journal of Mechanical Design*, 139(11):111408, 2017. 2

[10] Y. Du, Z. Liu, H. Basevi, A. Leonardis, B. Freeman, J. Tenenbaum, and J. Wu. Learning to exploit stability for 3d scene parsing. In S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada.*, pages 1733–1743, 2018. 1

[11] M. Edmonds, F. Gao, X. Xie, H. Liu, S. Qi, Y. Zhu, B. Rothrock, and S.-C. Zhu. Feeling the force: Integrating force and pose for fluent discovery through imitation learning to open medicine bottles. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3530–3537. IEEE, 2017. 2

[12] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 1

[13] C. Fletcher. *Computational techniques for fluid dynamics., 2nd Edition*. Springer, 1991. 1

[14] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. Dobkin. Modeling by example. In *ACM transactions on graphics (TOG)*, volume 23, pages 652–663. ACM, 2004. 2

[15] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016. 2

[16] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014. 2

[17] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 1

[18] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1

[19] Y. He, X. Liu, H. Zhong, and Y. Ma. Addressnet: Shift-based primitives for efficient convolutional neural networks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1213–1222. IEEE, 2019. 1

[20] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang. Bounding box regression with uncertainty for accurate object detection. In *2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019. 1

[21] H. Jasak, A. Jemcov, and Z. Tukovic. Openfoam: A c++ library for complex physics simulations. 11 2013. 1, 2

[22] A. Javadi, T. Tan, and M. Zhang. Neural network for constitutive modelling in finite element analysis. *Computer Assisted Mechanics and Engineering Sciences*, 10(4):523–530, 2003. 2

[23] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[24] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 1

[26] L. Ladicky, S. Jeong, N. Bartolovic, M. Pollefeys, and M. Gross. Physicsforests: real-time fluid simulation using machine learning. In *ACM SIGGRAPH 2017 Real Time Live!*, pages 22–22. ACM, 2017. 2

[27] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In M. Balcan and K. Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 1558–1566. JMLR.org, 2016. 2, 3

[28] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436, 2015. 1

[29] K. Li, T. Pham, H. Zhan, and I. Reid. Efficient dense point cloud object reconstruction using deformation vector fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 497–513, 2018. 1

[30] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. 2

[31] O. Litany, A. Bronstein, M. Bronstein, and A. Makadia. Deformable shape completion with graph convolutional autoencoders. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1886–1895, 2018. 1

[32] S. Liu, L. Giles, and A. Ororbia. Learning a hierarchical latent-variable model of 3d shapes. In *2018 International Conference on 3D Vision (3DV)*, pages 542–551. IEEE, 2018. 2

[33] P. Mandikal, K. L. Navaneet, M. Agarwal, and R. V. Babu. 3D-LMNet: Latent embedding matching for accurate and diverse 3d point cloud reconstruction from a single image. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018. 1

[34] R. Mottaghi, M. Rastegari, A. Gupta, and A. Farhadi. "what happens if..." learning to predict the effect of forces in images. In *European Conference on Computer Vision*, pages 269–285. Springer, 2016. 2

[35] S. Oh, Y. Jung, I. Lee, and N. Kang. Design automation by integrating generative adversarial networks and topology optimization. In *ASME 2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages V02AT03A008–V02AT03A008. American Society of Mechanical Engineers, 2018. 2

[36] L. Pinto, D. Gandhi, Y. Han, Y.-L. Park, and A. Gupta. The curious robot: Learning visual representations via physical interactions. In *European Conference on Computer Vision*, pages 3–18. Springer, 2016. 2

[37] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 77–85, 2017. 2

[38] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 5105–5114, 2017. 2

[39] G. Riegler, A. Osman Ulusoy, and A. Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017. 2

[40] J. Schulman, S. Levine, P. Abbeel, M. I. Jordan, and P. Moritz. Trust region policy optimization. In F. R. Bach and D. M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1889–1897. JMLR.org, 2015. 2

[41] C. Settgast, M. Abendroth, and M. Kuna. Constitutive modeling of plastic deformation behavior of open-cell foam structures using neural networks. *Mechanics of Materials*, 131:1–10, 2019. 2

[42] T. E. Tezduyar. Stabilized finite element formulations for incompressible flow computations. In *Advances in applied mechanics*, volume 28, pages 1–44. Elsevier, 1991. 5

[43] R. W. Thatcher. *Theory and application of the finite element method*. PhD thesis, Imperial College London, UK, 1971. 1

[44] J. Tompson, K. Schlachter, P. Sprechmann, and K. Perlin. Accelerating eulerian fluid simulation with convolutional networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3424–3433. JMLR. org, 2017. 2

[45] S. Tulsiani, A. A. Efros, and J. Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *Computer Vision and Pattern Regognition (CVPR)*, 2018. 1

[46] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik. Learning shape abstractions by assembling volumetric primitives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2635–2643, 2017. 1

[47] N. Umetani. Exploring generative 3d shapes using autoencoder networks. In D. Gutierrez and H. Huang, editors, *SIGGRAPH Asia 2017 Technical Briefs, Bangkok, Thailand, November 27 - 30, 2017*, pages 24:1–24:4. ACM, 2017. 2, 4, 5

[48] N. Umetani and B. Bickel. Learning three-dimensional flow for interactive aerodynamic design. *ACM Transactions on Graphics (TOG)*, 37(4):89, 2018. 2

[49] N. Umetani, Y. Koyama, R. Schmidt, and T. Igarashi. Pteromys: interactive design and optimization of free-formed free-flight model airplanes. *ACM Transactions on Graphics (TOG)*, 33(4):65, 2014. 2

[50] O. Van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A survey on shape correspondence. In *Computer Graphics Forum*, volume 30, pages 1681–1707. Wiley Online Library, 2011. 2

[51] N. Verma, E. Boyer, and J. Verbeek. Feastnet: Feature-steered graph convolutions for 3d shape analysis. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 2598–2606, 2018. 3

[52] J. Wang, L. Qian, E. Azimi, and P. Kazanzides. Prioritization and static error compensation for multi-camera collaborative tracking in augmented reality. In *2017 IEEE Virtual Reality (VR)*, pages 335–336. IEEE, 2017. 1

[53] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 1

[54] P.-S. Wang, Y. Liu, Y.-X. Guo, C.-Y. Sun, and X. Tong. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Transactions on Graphics (TOG)*, 36(4):72, 2017. 2

[55] Z. Wang, S. Rosa, L. Xie, B. Yang, S. Wang, N. Trigoni, and A. Markham. Defo-net: Learning body deformation using generative adversarial networks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2440–2447. IEEE, 2018. 1

[56] Z. Waszczyszyn and L. Ziemiański. Neural networks in mechanics of structures and materials–new results and prospects of applications. *Computers & Structures*, 79(22-25):2261–2276, 2001. 2

[57] D. C. Wilcox et al. *Turbulence modeling for CFD*, volume 2. DCW industries La Canada, CA, 1998. 5

[58] J. Wu, E. Lu, P. Kohli, B. Freeman, and J. Tenenbaum. Learning to see physics via visual de-animation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 153–164. Curran Associates, Inc., 2017. 1

[59] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017. 1

[60] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 82–90, 2016. 2

[61] X. Wu, J. Ghaboussi, and J. Garrett Jr. Use of neural networks in detection of structural damage. *Computers & structures*, 42(4):649–659, 1992. 2

[62] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 1, 2

[63] M. E. Yumer, S. Chaudhuri, J. K. Hodgins, and L. B. Kara. Semantic shape editing using deformation handles. *ACM Transactions on Graphics (TOG)*, 34(4):86, 2015. 2

[64] C. Zhu, Y. He, and M. Savvides. Feature selective anchor-free module for single-shot object detection. *arXiv preprint arXiv:1903.00621*, 2019. 1

[65] Y. Zhu, C. Jiang, Y. Zhao, D. Terzopoulos, and S.-C. Zhu. Inferring forces and learning human utilities from videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3823–3833, 2016. 2