LADIS: Language Disentanglement for 3D Shape Editing

Ian Huang[†], **Panos Achlioptas**[§], **Tianyi Zhang**[†],
Sergey Tulyakov[§], **Minhyuk Sung**[‡] and **Leonidas Guibas**[†]

[†]Department of Computer Science, Stanford University
{ianhuang, tianyizhang, guibas}@cs.stanford.edu

[‡] School of Computing, KAIST
mhsung@kaist.ac.kr

[§]Snap Research
{panos, stulyakov}@snap.com

Abstract

Natural language interaction is a promising direction for democratizing 3D shape design. However, existing methods for text-driven 3D shape editing face challenges in producing *decoupled, local edits* to 3D shapes. We address this problem by learning disentangled latent representations that ground language in 3D geometry. To this end, we propose a complementary tool set including a novel network architecture, a disentanglement loss, and a new editing procedure. Additionally, to measure edit locality, we define a new metric that we call *part-wise edit precision*. We show that our method outperforms existing SOTA methods by 20% in terms of edit locality, and up to 6.6% in terms of language reference resolution accuracy. Human evaluations additionally show that compared to the existing SOTA, our method produces shape edits that are more local, more semantically accurate, and more visually obvious. Our work suggests that by solely disentangling language representations, downstream 3D shape editing can become more local to relevant parts, even if the model was never given explicit part-based supervision.

1 Introduction

Natural language has been used as a universally accessible and powerful interface to enable content creation and manipulation. For 2D images, Dall·E (Ramesh et al., 2022), Imagen (Saharia et al., 2022a), and StyleCLIP (Patashnik et al., 2021) demonstrated impressive capabilities in creating and editing images using language descriptions.

However, understanding and grounding natural language in 3D geometry remains a challenge. Recent efforts have all focused on using existing large pretrained vision-language models like CLIP (Radford et al., 2021), along with a differentiable renderer to interface 3D geometry with CLIP through 2D renderings. While these methods can perform geometry edits as described by language, they often also create unintended, additional, and unwanted changes. For example in Figure 1, while the edit instruction asks for the chair legs to become thinner, existing methods can produce spurious editing artifacts elsewhere, like the back of the chair. To minimize the artifacts created by these changes, works such as ChangeIt3D (Achlioptas et al., 2022) often regularize their predicted edits to be very conservative, resulting in the overly subtle changes in regions of the shape where edits are requested, as shown in Figure 1. Such edit behaviors can be counter-intuitive and detrimental to the editing experience, interfering with the design process.

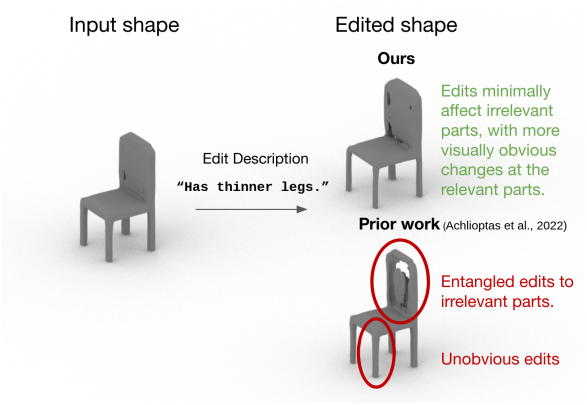


Figure 1: An example of the language-conditioned 3D shape editing task. Given an input shape and an edit description, the objective is to edit the shape to match the edit description. When language representations are entangled, this can often produce edits not localized to the relevant parts. In contrast, by regularizing the disentanglement of language representation *alone*, our system achieves higher edit locality while making more pronounced edits in the requested areas.

We hypothesize that the source of non-local edits lies in the way we jointly represent edit descriptions and shapes. To perform language-based shape edits, existing methods first learn a shape-language joint space using a classification or contrastive learning objective. Then, the system iteratively perform shape edits to maximize the similarity between the

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 5519–5532
December 7-11, 2022

final shape representation and the edit instruction representation (Michel et al., 2022; Wang et al., 2022; Hong et al., 2022; Jain et al., 2022). Problems arise when the joint representation space is entangled, i.e., the representations of independent edit instructions referring to unrelated shape edits have non-trivial co-dependencies. Optimizing with such entangled language representations can cause unintended correlations in unrelated shape edits.

In this work, we study how to achieve *localized* language-based 3D shape editing through learning more disentangled language representations. Given an input shape and an utterance, the system should perform the correct edits to the shape while minimizing edits to parts of the shape that are unnecessary to shape integrity and undescribed by the edit description.

Thus, we aim to learn *disentangled* grounded language representations that link language references to the correct parts of the shape. In an ideal disentangled latent space, *independent* edit descriptions that describe mutually independent attributes and parts should be orthogonal to each other (e.g. "the car has big wheels" and "the car has 2 doors"). Figure 2 shows an example of two independent edit descriptions.

To this end, we propose a language disentanglement loss, named LADIS, and a multi-expert shape-difference encoder architecture. Whereas the LADIS loss enables the model to orthogonalize mutually independent utterances, the multi-expert shape-difference encoder allows the experts to specialize on different geometric features that span the semantics of the edit descriptions. These two design decisions enable our model to decouple different adjectives and part references within the edit descriptions, leading to improved disentanglement.

We note that a certain degree of geometry entanglement may be inherent in the integrity of the shape structure – the use of different linguistic terms (e.g., "table top", "table leg") does not always imply perfect disentanglement of the part locations or geometries. For example, we typically expect co-dependencies between the extent of a four-leg table top and its leg locations. Such dependencies have to be learned from data and encoded in the shape space, making our task quite challenging.

Even with a more disentangled joint representation space, imperfect optimization makes shape editing difficult. Empirically, we find that optimization-based 3D editing approaches may generate invalid objects. For example, failed shape edits can cause a chair shape to stop looking like a realistic chair. To combat this issue, we additionally introduce two modifications to a standard optimization procedure: neighborhood simplex editing (NSE) and output-driven edit step-size adjustment (ODESSA). NSE re-frames shape editing as interpolating between a source shape and its nearest neighbors, which helps the edited shape to remain realistic. ODESSA enables every optimization step to produce a constant amount of change in the output 3D space, allowing us to cope with metric ambiguities in language instructions.

We evaluate our system on the ShapeTalk (Achlioptas et al., 2022) and ShapeGlot (Achlioptas et al., 2019) datasets. To quantify the degree of edit locality, we propose *part-wise edit precision* (PEP), a new metric that measures if shape edits are restricted to the parts that are mentioned by the edit description. Experimental results show that our system demonstrates an improved ability to resolve language references and successfully produce more local shape edits (20% higher PEP than the closest competing system (Achlioptas et al., 2022)). Additionally, human evaluation comprising of 500 annotations shows that compared to ChangeIt3D (Achlioptas et al., 2022), our method produces 3D shape edits with higher part-wise locality, semantic accuracy and visual obviousness. Furthermore, our analysis of the language representations shows a higher level of disentanglement of shape part concepts and adjectives, demonstrating that disentangling language representation enables larger, more correct, and more localized shape edits, without ever equipping our model with explicit part supervision. The code for LADIS can be found at `https://github.com/ianhuang0630/LADIS`.

2 Related Work

2.1 Language Grounding in 3D shapes

Prior works (Achlioptas et al., 2019, 2022; Thomason et al., 2021) study language grounding in the *geometry* of common 3D objects by first collecting natural referential language, extracted by deploying referential games (Lewis, 1969; Kazemzadeh et al., 2014) that use 3D shapes as visual grounding. Then, tapping on such visio-linguistic data the aforementioned works and similar ones (Achlioptas et al., 2020; Chen et al., 2019; Anonymous, 2023;

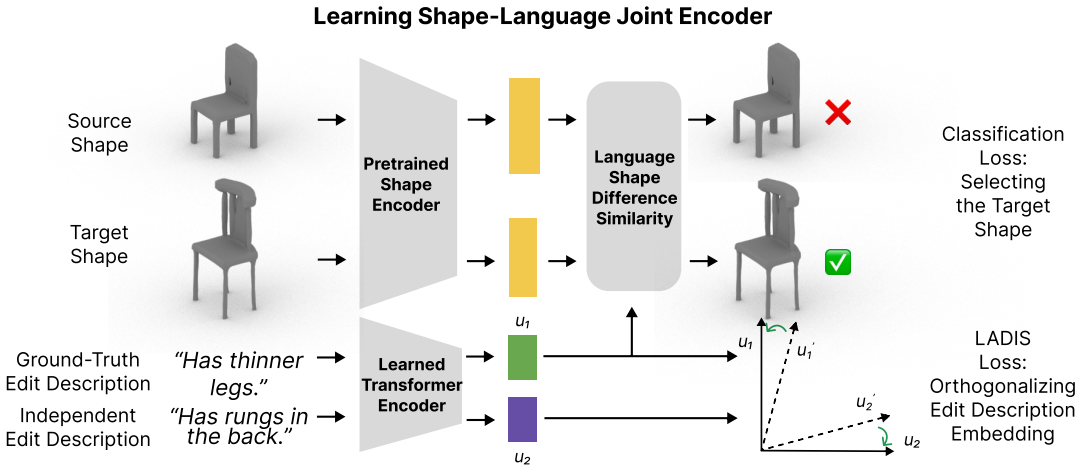


Figure 2: Overview of the training for a disentangled language-shape difference joint space. Our network solves a binary classification task of determining which input shape is the target, and in this process must jointly reason about both the shape differences and the edit description. To disentangle this space, we regularize the process with the LADIS loss, which encourages independent instructions to be orthogonalized within the joint space.

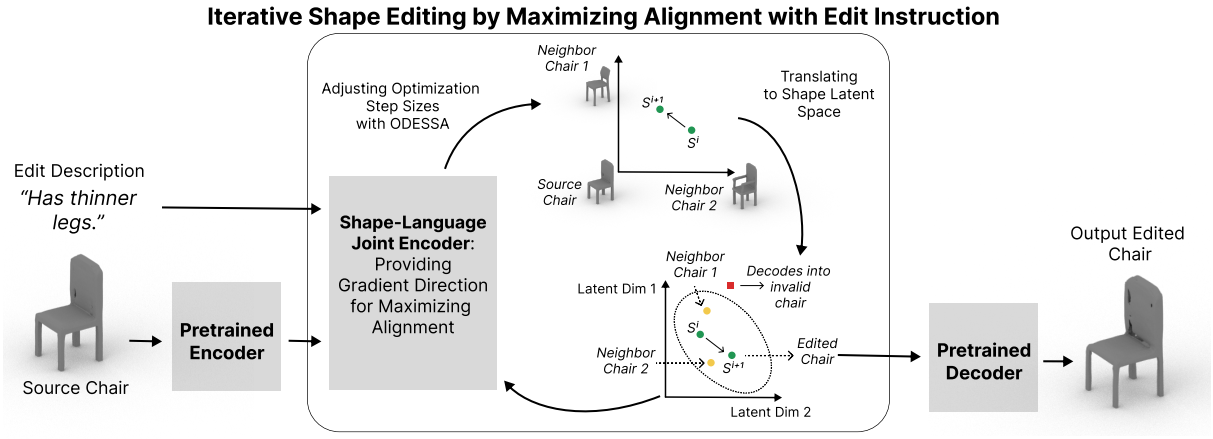


Figure 3: Overview of the editing procedure. The system first retrieves a set of nearest neighbors to create an coordinate space in which our edited shape will be expressed. Using the joint space $\mathcal{J}$, gradient steps iteratively increase the semantic similarity of the edited shape difference and the edit description, modulated by ODESSA. The pretrained decoder can be used to decode the final edited latent code to an output.

Koo et al., 2022), train deep-learning-based models by learning to solve 3D-grounded referential tasks.

While our work focuses on learning disentangled representations to edit shapes, we also show that our proposed method can lead to improvements in language reference resolution as well. ChangeIt3D (Achlioptas et al., 2022) introduced the language-driven shape editing task and contributed a large scale dataset (ShapeTalk) to enable data-driven solutions for it. Here, we improve upon baseline methods introduced in ChangeIt3D. Methods like DreamFusion (Poole et al., 2022), DreamFields (Jain et al., 2022), CLIPNerf (Wang et al., 2022), Text2Mesh (Michel et al., 2022) and AvatarCLIP (Hong et al., 2022) differentiably render 3D objects into images and use pretrained vision-language models such as CLIP (Radford et al., 2021) or pretrained text-to-image models like Imagen (Saharia et al., 2022b) to both linguistically ground and optimize the geometry to semantically align with the language input.

While these are promising approaches, their value is mainly in generating new shapes from scratch, where the idea of "edit locality" no longer applies. Nonetheless, the question of how to produce minimal, localized edits to a shape remains an important one, if text-guided shape editing is to be widely adopted in the future for 3D content creation

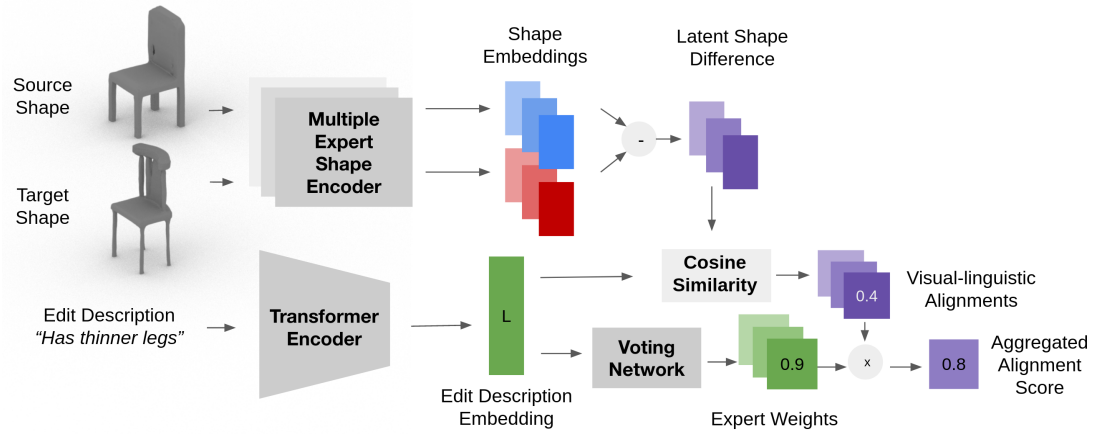


Figure 4: The architecture used to learn a disentangled joint space for language and shape differences. Note that multiple expert models are used to represent shape differences in different spaces, and the corresponding similarity predictions are aggregated by a voting network that chooses among these spaces based on the sentence embedding.

pipelines. As such, our work studies grounding language in pure 3D representations, and specifically aims to produce *disentangled* shape edits.

2.2 Disentangled Representations

In the deep learning literature, disentangled representations have been studied mostly for learning generative models. InfoGAN (Chen et al., 2016) pioneered this line of research while modifying a GAN objective function to maximize the mutual information between latent codes and the generated data. β-VAE (Higgins et al., 2017) extended this idea with a regularization loss enforcing statistical independence of the latent factors, which enabled more stable training and the use of fewer assumptions on the data distribution. Follow-up work such as CorEx (Gao et al., 2019), β-TCVAE (Chen et al., 2018b), and FactorVAE (Kim and Mnih, 2018) proposed information-theoretic approaches to resolve this issue with a total correlation penalty, and a generalization of the mutual information metric for multivariate cases. HFVAE (Esmaeili et al., 2019) also modified the objective function to achieve hierarchical factorization, and DIP-VAE (Kumar et al., 2018) introduced a different regularization based on covariance penalization.

For 3D shape editing, there is little work on learning disentangled representations. Aumentado Armstrong et al. (Aumentado-Armstrong et al., 2019) were the first to leverage the ideas of HFVAE (Esmaeili et al., 2019) and DIP-VAE (Kumar et al., 2018) on 3D shape data. The approach was extended in DeepMetaHandles (Liu et al., 2021) to learn keypoint-based intuitive meta-handles. DeformSyncNet (Sung et al., 2020) also proposed learning a dictionary for editing, whose disentanglement is enforced with a sparsity regularization over the per-point offsets of the deformation. Compared to these works, we introduce a novel method for learning disentangled *language* representation to edit 3D shapes given a language prompt.

3 Method

3.1 Problem Statement

We are interested in the language-based shape editing task: given a source shape S and edit description u, we want to modify S to fulfill the edits mentioned in u. Our model learns to accomplish this task by learning on a dataset of edit descriptions describing the *difference* between some source shape S and target shape T. Importantly, as shown in Figure 1, u is not expected to describe *all* the characteristics of either S or T, just some ways in which T is *different* from S. S and T can be given to us in various 3D representations (meshes, implicits, etc.). We assume access to pretrained shape autoencoders that can both encode to and decode from latent vector representations of shapes. In other words, $s = enc(S) \in \mathbb{R}^d$ is a vector representing the 3D shapes S and $S = dec(s)$ represents the inverse operation.

The key to tackle the language-based shape editing task is linking an edit instruction u with the shape difference between the target T and source S, which we denote as $diff(S, T)$. We achieve this by embedding $diff(S, T)$ and u into the same vector space $\mathcal{J}$, as other works like Text2Shape (Chen et al., 2018a) and CLIP (Radford et al., 2021) have done.

Inspired by prior work studying shape differences (Mo et al., 2020), we represent *diff*(S, T) as first-class citizens. Specifically, we model *diff*$(S, T) \in \mathcal{J}$ as $f(t) - f(s)$, where $f(x) \in \mathcal{J}$ is a shape encoder on top of the encoded latent representation x of the input shape. To project the edit instruction to $\mathcal{J}$, we use encoder $g(u) \in \mathcal{J}$. We learn f and g so that the cosine similarity cossim$(f(t) - f(s), g(u))$ is high. Equipped with $\mathcal{J}$, we then can perform editing using optimization. We initialize s' as a perturbed version of s and iteratively change s' based on u to maximize sim$(f(s') - f(s), g(u))$. During editing, we keep the parameters of f, g unchanged and only keep updating s'. We refer these two stages as **Grounded Representation Learning** and **Optimization-based Editing**. We now discuss these two stages respectively.

3.2 Grounded Representation Learning

In order to learn a good grounded representation space $\mathcal{J}$, we propose to learn a classification task. Given the triplet (s, t, u), we train the encoders f and g by learning a binary classification task of identifying whether s or t is the correct edit target,

$$\mathcal{L}_{\text{binary}}(s, t, u) = \\ -\log\left(\frac{\exp h(s,t,u)}{\exp h(s,t,u) + \exp h(t,s,u)}\right)$$

where $h(s, t, u) = \text{sim}(f(t) - f(s), g(u))$.

While L_{binary} is useful for learning grounded representations, it does not encourage disentangling the representations in $\mathcal{J}$. In order to obtain a disentangled joint space $\mathcal{J}$, we need to disentangle the representations of both the shape and the edit instruction. As shown in Figure 2, we first design a multi-expert network architecture that separates each shape representation into multiple specialized vectors and then propose a language disentanglement loss that orthogonalizes representations of independent edit instructions. In the next two sections, we discuss these two techniques respectively.

3.2.1 Multi-Expert Shape Difference Encoder

There are many different perspectives on describing a 3D shape, including dimension, texture, structure, etc. Ideally, we can represent each aspect of a shape with a different representation. We instantiate this intuition by designing a multi-expert architecture (Figure 4), where a collection of experts $f_1, f_2 ... f_k$ project the input shapes into separate spaces. For an expert f_i, we define the shape difference as $f_i(t) - f_i(s)$. We aggregate each of the projected shape differences via a voting network $w(u) = \text{Softmax}(MLP(g(u)))$, by:

$$f(t) = \sum_{i=1}^{k} w_i(u) f_i(t). \quad (1)$$

With f as a mapping from each shape into $\mathcal{J}$, we measure the alignment $h(s, t, u)$ between the shape difference and edit instruction by cosine similarity with the encoding of the edit instruction representation $g(u)$,

$$h(s, t, u) = \text{cossim}(g(u), f(t) - f(s)). \quad (2)$$

In practice, we implement g with a transformer encoder (Vaswani et al., 2017) and take the representation of the first token. We implement each shape encoder expert f_i as an MLP ontop of the latent shape representation from a pretrained shape autoencoder. We additionally make the temperature parameter τ for $w(u)$ a learnable parameter, which empirically lowers throughout training automatically as the experts specialize.

3.2.2 Language Disentanglement Loss

Recall that our end goal of learning a disentangled joint space $\mathcal{J}$ is to ensure localized edits: when the user asked for making the legs thinner, the system should not add holes to the back of the source chair (Figure 1). In our framework, this intuition translates to ensuring that independent edit instructions u and u^- have zero similarity, i.e. cossim$(g(u), g(u^-)) \approx 0$.

With this goal in mind, we design a regularization loss by mining independent edit instructions. We use $M(u) = \{u_1^-, u_2^-, ...u_k^-\}$ to represent the action of retrieving independent instructions for u. With these independent edit instructions, we directly optimize for our previous intuition by proposing a language disentanglement (LADIS) loss,

$$\mathcal{L}_{\text{LADIS}}(u) = \sum_{u^- \in M(u)} |g(u) \cdot g(u^-)|. \quad (3)$$

In practice, we design different sampling strategies for mining independent edit instructions based on the dataset structure. We optimize the shape difference encoder by adding $\mathcal{L}_{LADIS}$ and $\mathcal{L}_{\text{binary}}$ and tune the ratio between them as a hyperparameter.

Algorithm 1 Calculate edited latent s'

Require: input shape s, edit utterance u,
Require: shape decoder *dec*, Number of nearest neighbors $P > 0$, Variance $\gamma > 0$, Expected output change $\delta > 0$, B optimization steps.

1: $Q \leftarrow \text{GetNearest}(P, s)$
2: $\epsilon \sim N(0, \gamma)$
3: $s' \leftarrow s + \epsilon^T Q$
4: **for** B steps **do**
5: $\quad \Delta\epsilon \leftarrow \nabla_\epsilon h(s, s', u)$
6: $\quad \eta \leftarrow \textit{ODESSA}(s', \Delta\epsilon, \delta)$
7: $\quad \epsilon \leftarrow \epsilon + \eta\Delta\epsilon$
8: $\quad s' \leftarrow s + \epsilon^T Q$
9: **end for**
10: **return** s'

3.3 Shape Editing with Iterative Optimization

While a disentangled shape-language joint space $\mathcal{J}$ provides strong signals, we now discuss the algorithm to convert these signals to actual shape edits. Recall that we encode the input shape S using a pretrained auto-encoder into s. As shown in Figure 3, we follow the standard approach to iteratively traverse the latent encoder space to find a new shape s' that maximizes $h(s, s', u)$ and then decoded s' with the pretrained decoder (Algorithm 1).

However, the encoder latent space is nonlinear and a naive optimization often leads to degenerate edits. To better guide the latent space traversal, we propose two modifications: we adjust the direction of traversal by leverage a valid object neighborhood and adjust the traversal step size by controlling for the total change incurred by the edit in the output.

3.3.1 Neighborhood Simplex Editing (NSE)

To prevent the latent space traversal from deviating off the valid shape manifold, we retrieve P nearest neighbors from the training set using s, forming a P-simplex $Q \in \mathbb{R}^{P \times d}$ within a higher-dimensional latent space (Algorithm 1, Line 6). We constrain the traversal by treating Q as base coordinates, and optimizing s' with these new coordinates by setting $s' = s + \epsilon^T Q$, as shown in Figure 3. Our intuition is that P can be chosen well enough to cover most modes of variations from the source shape s, while being low-dimensional enough (in comparison to the latent space dimension d) to constrain the editing to happen on the shape manifold. Note that these modes of variation do not necessarily have to be captured in a "positive" directional sense. For example, since the elements of ϵ are allowed to be negative, for the utterance "shorter legs", a neighboring chair within the chair space with *longer* legs than the input shape can be equally as informative as one with *shorter* legs. Ablations involving NSE is included in the Appendix.

3.3.2 Output-Driven Edit Step-Size Adjustment (ODESSA)

The high-dimensional nonlinear nature of the encoder latent space makes editing challenging because a small step in the latent space can potentially cause both catastrophically large changes as well as unnoticeable changes in the decoded shape. To combat this, we rescale the size of each edit step so that the total change in the decoded output is similar. We denote this operation as *ODESSA*$(s', \Delta\epsilon, \delta)$, which outputs a scalar as the step size in the direction $\Delta\epsilon$ based on an expected total change δ in the output space. We implement *ODESSA* by calculating the changes on the decoded 3D representation and present the implementation details and relevant ablations in the Appendix.

4 Experiments

In this section, we evaluate our system's ability to achieve localized language-guided shape edits. We evaluate our disentangled joint space $\mathcal{J}$ by shape classification accuracy, and by the locality of our system's edits, evaluated both by our novel part-based metric and human evaluators. In addition, we analyze expert specialization in our multi-expert shape difference encoder and visualize the disentangled language representations. Experimental results show that our proposed method is better at shape classification, produces more disentangled representations, and lead to more local shape edits.

4.1 Experiment Setup

4.1.1 Datasets

There are two datasets that provide textual information for differences between shapes: ShapeTalk (Achlioptas et al., 2022) and ShapeGlot (Achlioptas et al., 2019). In both, text describes a distinguishing feature that a target shape has which the source shape(s) do not. Within ShapeTalk, each sample has only a single source shape, but multiple separate lines of text are collected from human labelers that enumerate many differences between the source and the target. For ShapeGlot, the text is more pragmatic, and used to differentiate a target from *two* source shapes. This

also means that a single textual description can very sparsely describe the difference between the source and the target. In our experiments, we use these textual descriptions as edit descriptions.

4.1.2 Evaluating Metrics

In order to quantify the degree to which our edits are local to certain parts, we introduce **Part-based Edit Precision** (PEP), which is defined as the log of the ratio between the percentage volume change found in the "relevant" regions of the input 3D shape, versus the entire shape. For a subset of chairs, we manually associate certain keywords within edit descriptions to Partnet (Mo et al., 2019) part labels, and this enables us to measure the percentage volume change observed in the regions associated with mentioned parts. The PEP score is 0 when the percentage volume change in the mentioned area is the same as the percentage volume change across the whole entire shape. Thus, higher PEP indicates that an edit has high locality. We additionally measure Δv, the total change in volume. Further information about PEP and Δv is provided in the Appendix.

In what follows, we evaluate the mean PEP (mPEP) score and mean Δv (mΔv) for 2000 chairs within the ShapeTalk test set with existing PartNet (Mo et al., 2019) part segmentations.

4.2 Experimental Results

4.2.1 Shape Editing

We investigate whether the disentangled language representations allow for more localized edits. We compare against the SOTA method (Achlioptas et al., 2022) on the ShapeTalk dataset using mPEP.

Latent Geometric Representations In our editing experiments, we use the IM-NET (Chen and Zhang, 2019) as a shape autoencoder, as it enables easy conversion to meshes, and therefore enables better comparisons of edit locality.

Mining for independent utterances For ShapeTalk, labelers are prompted to enumerate different shape differences for every source-target pair. As such, the utterances provided for a single pair by a single labeler are very likely independent. We can thus instantiate $M(u)$ as the multiple utterances gathered from a single labeler for the same source-target pair (**multiutterance LADIS**). As a compromise in settings where labeler information is not available, we instantiate $M(u)$ as the other edit instructions within the dataset used to describe the *same source-target pair* (**shared-context LADIS**).

	Multiutt	ShareCon	ChangeIt3D
mPEP ↑	**0.378**	0.338	0.315
mΔV ↑	0.0334	**0.0416**	0.0105

Table 1: Comparison of edit locality (mPEP) and mean edit volume (mδV) with the ChangeIt3D baseline on ShapeTalk. For multiutterance LADIS (multiutt) and shared-context LADIS (ShareCon), our models achieve higher locality and more volumetrically obvious edits.

	Multiutt	ShareCon	ChangeIt3D
IM-NET	**69.52%**	69.03%	62.9 %
ShapeGF	70.62%	**70.94%**	65.64 %

Table 2: Source-target classification accuracies of multiutterance LADIS (Multiutt) and shared-context LADIS (SharCon) on ShapeTalk, compared to ChangeIt3D (Achlioptas et al., 2022). Both versions of our models outperform ChangeIt3D, on both types of latent representations.

Experimental Results Table 1 compares our method with ChangeIt3D. We find that LADIS not only produce edits that are 20% more localized (measured by mPEP), but also volumetrically larger changes ($\geq 3\times$ that of ChangeIt3D edits), allowing our system to achieve more visually pronounced and localized shape edits than ChangeIt3D edits, as shown in the sample in Figure 1. We perform a dependent sample T-test between the ChangeIt3D baseline and our method (specifically the multiutterance variant) and confirm that the PEP improvement was statistically significant (p=0.0287). This verifies our hypothesis that learning disentangled representations can better ensure the locality of the geometric edits, even though no explicit part information was ever given to the model.

4.2.2 Language Reference Resolution

We next measure the quality of the shape-language joint space by evaluating the source-target shape binary classification accuracy, which is the training objective of shape difference encoder.

Latent Shape Representations In addition to IM-NET, we experiment with the ShapeGF (Cai et al., 2020) autoencoder, which learns to reconstruct shapes by deforming an initial prior distribution of points according to a gradient field.

Mining for independent utterances As an additional compromise where the set of utterances

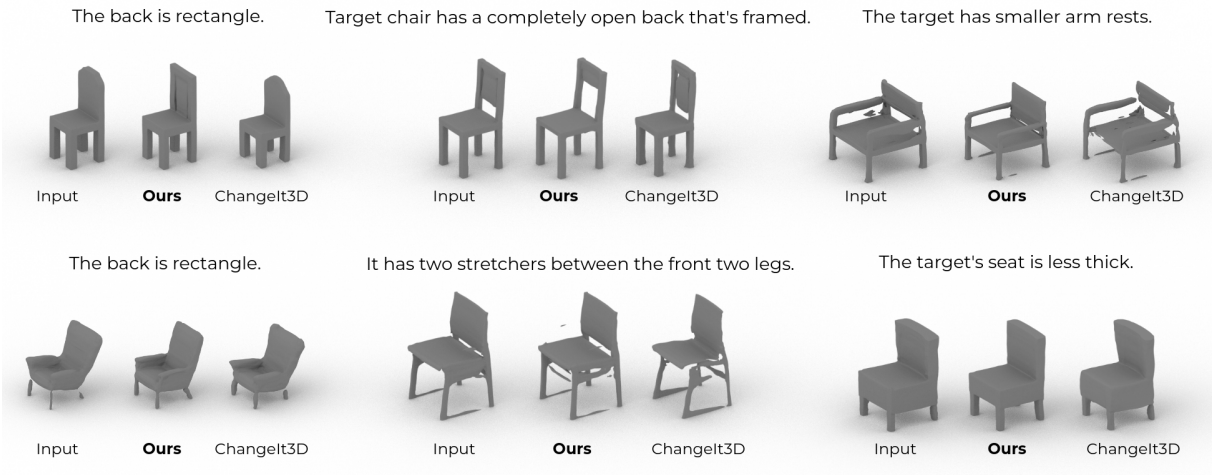


Figure 5: Qualitative examples of our method editing different kinds of chairs, compared to ChangeIt3D. The text *describing the desired output* is shown above the samples. Compared to ChangeIt3D, our method allows for more semantically correct and obvious edits, while improving the locality of the shape edit.

	Random LADIS	w/o LADIS
IM-NET	**78.43%**	76.15%
ShapeGF	**76.58%**	75.59%

Table 3: Source-target binary classification accuracies on ShapeGlot. LADIS loss applied on even random utterances yield an notable improvement in accuracy for both IM-NET and ShapeGF latent representations.

per source-target is prohibitvely sparse, we explore using utterances from different source-target pairs within the batch to be $M(u)$ (**random LADIS**). We do this with ShapeGlot, and break every triplet of shapes into two pairs $(\text{source1}, \text{target})$ and $(\text{source2}, \text{target})$ for training and testing.

Experimental Results Table 3 and Table 2 show that models trained with the LADIS loss has a notable increase in classification accuracy across two datasets and two backbone networks. Table 3 also reveals that even when loosely independent edit instructions are used for LADIS loss, we still observe a performance gain on the ShapeGlot dataset. Our findings show that LADIS produces higher quality language-shape joint spaces J, which may explain its effectiveness for downstream shape editing.

4.2.3 Human Evaluation

Figure 5 displays some examples of our method's shape edits on the chair category. Qualitative results show that on average, our method produces more semantically accurate and localized edits.

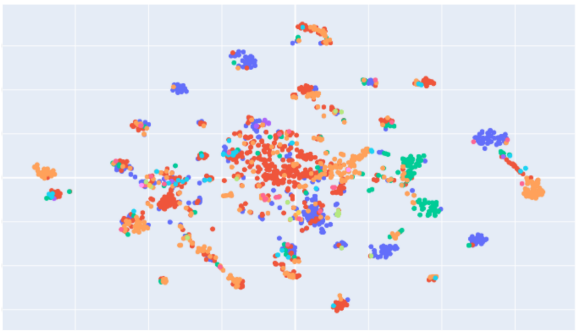
To evaluate this more systematically, we conduct a human evaluation comparing our method and ChangeIt3D. Our evaluation employs 10 volunteer annotators from our institute and in total, has 500 annotations. Our annotators are given the input, a language description of the desired output, and a pair of edited outputs (from our method and ChangeIt3D), and are tasked with deciding which of the edits: (1) is most obviously aligned with the language instruction and (2) is the most localized (i.e. preserves the identity of the unmentioned parts of the object). More details about the annotation process is detailed in the Appendix.

In 72% of the cases, our edits were deemed more semantically accurate (more semantically aligned and visually obvious) compared to ChangeIt3D. In 61% of the cases, our results were deemed more localized. Looking across the evaluators, all 10 evaluators unanimously judge that our method produces more accurate and obvious edits than ChangeIt3D. 8 of the 10 evaluators judge that our method produces strictly more localized edits than ChangeIt3D, with the other 2 evaluators indicating a tie.

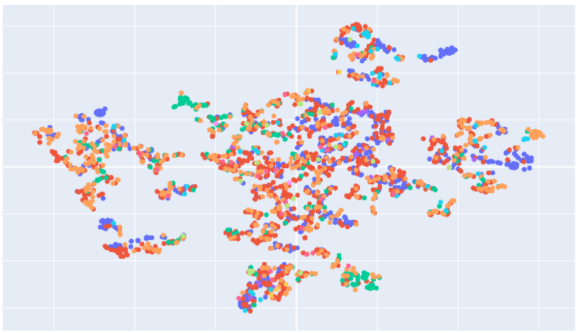
In 65% of the cases, a higher PEP score directly corresponds to a subjective perception of higher locality of edits reported by human evaluators. This also motivates the use of PEP as an automatic evaluation metric.

4.2.4 Visualizing Disentanglement

What is the effect of the LADIS loss on the language representations? Figure 6 shows a T-SNE embedding of a subset of edit descriptions within the joint space $\mathcal{J}$. We color-code edit description based on the various chair parts that are mentioned.

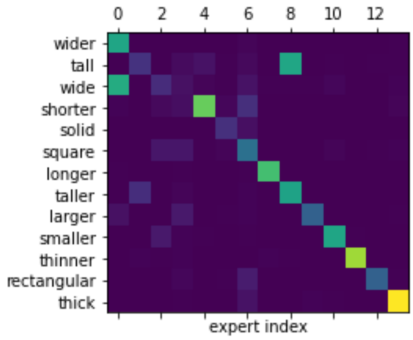


(a) Multiutterance LADIS

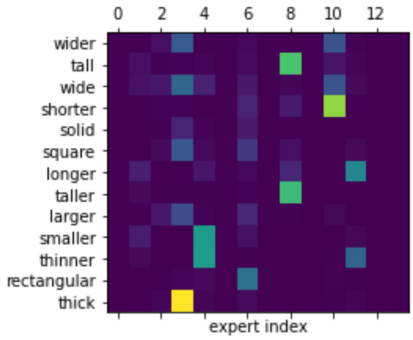


(b) Without LADIS

Figure 6: TSNE visualization of the learned language embeddings for **(a)** multi-utterance LADIS and **(b)** our model without LADIS loss, with the colors indicating the sets of parts mentioned within each utterance.



(a) Multiutterance LADIS



(b) Without LADIS

Figure 7: Visualizing the activation of different experts for different utterances reveals that the LADIS loss allows for different experts activate for different sets of adjectives. Notice that synonymous words like "wider"/"wide" and "tall"/"taller" are allocated to the same experts. Compare to the baseline without LADIS loss, where the same level of specialization is not as clearly observed.

Comparing representations trained with LADIS to a baseline without LADIS, we see that representation clusters are more distinct and better correspond to certain part identities.

To analyze the effect of the LADIS loss on the shape representations, we visualize expert specialization. We group edit descriptions by the adjective that appear in them and aggregate expert weights $w(u)$. Recall that expert weights $w(u)$ decides the importance of each expert representation for different edit instruction. Figure 7 shows experts trained with LADIS loss specialized to different adjectives. For example, whenever an edit description contains the word "longer", the seventh expert is weighted most heavily. In contrast, without the LADIS loss, expert specialization occurs to a much lesser extent. In conclusion, our representation analysis suggests that the LADIS loss helps disentangle both the language and the geometry representations.

5 Conclusion

In this work, we introduce a set of complimentary techniques to improve language-based shape edits. Our core idea is to learn a more disentangled shape-language joint space, which is achieved by a novel multi-expert architecture and a language disentanglement loss. Importantly, we show that we can accomplish more obvious edits while improving their locality by learning to disentangle the language *alone*, without ever equipping our model with part supervision. Experimental results show the effectiveness of our methods in producing localized shape edits. Our techniques are agnostic to the underlying 3D representations, and we look forward to extending our method to other 3D representations and modalities.

Limitations

In this work, we study the task of language-based shape editing. We limit the scope of our study to previous datasets, where only English is studied. We believe it's important future work to studying how other spoken and sign languages can be used for language-based shape editing.

Acknowledgements I. Huang and L. Guibas acknowledge the support of ARL grant (W911NF-21-2-0104), a Vannevar Bush Faculty Fellowship, and gifts from the Adobe and Snap Corporations. M. Sung acknowledges the support of NRF grant (2022R1F1A1068681) funded by the Korea government(MSIT) and grants from Adobe, KT, Samsung Electronics, and ETRI.

References

Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas J. Guibas. 2020. ReferIt3D: Neural listeners for fine-grained 3d object identification in real-world scenes. In *ECCV*.

Panos Achlioptas, Judy Fan, Robert D. Hawkins, Noah D. Goodman, and Leonidas J. Guibas. 2019. ShapeGlot: Learning language for shape differentiation. In *ICCV*.

Panos Achlioptas, Ian Huang, Minhyuk Sung, Sergey Tulyakov, and Leonidas Guibas. 2022. ChangeIt3D: Language-assisted 3d shape edits and deformations. https://changeit3d.github.io/.

Anonymous. 2023. 3D-Scene-Entities: Using phrase-to-3D-object correspondences for richer visio-linguistic models in 3D scenes. In *Submitted to The Eleventh International Conference on Learning Representations*. Under review, https://openreview.net/forum?id=coMWK6WGkBP.

Tristan Aumentado-Armstrong, Stavros Tsogkas, Allan Jepson, and Sven Dickinson. 2019. Geometric disentanglement for generative latent shape models. In *ICCV*.

Ruojin Cai, Guandao Yang, Hadar Averbuch-Elor, Zekun Hao, Serge Belongie, Noah Snavely, and Bharath Hariharan. 2020. Learning gradient fields for shape generation. In *ECCV*.

Kevin Chen, Christopher B Choy, Manolis Savva, Angel X Chang, Thomas Funkhouser, and Silvio Savarese. 2018a. Text2shape: Generating shapes from natural language by learning joint embeddings. In *ACCV*.

Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. 2018b. Isolating sources of disentanglement in vaes. In *NeurIPS*.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In *NeurIPS*.

Z. Dave Chen, Angel X. Chang, and Matthias Nießner. 2019. ScanRefer: 3D object localization in RGB-D scans using natural language. *arXiv*, abs/1912.08830.

Zhiqin Chen and Hao Zhang. 2019. Learning implicit fields for generative shape modeling. In *CVPR*.

Babak Esmaeili, Hao Wu, Sarthak Jain, Alican Bozkurt, N. Siddharth, Brooks Paige, Dana H. Brooks, Jennifer G. Dy, and Jan-Willem van de Meent. 2019. Structured disentangled representations. In *AISTATS*.

Shuyang Gao, Rob Brekelmans, Greg Ver Steeg, and A. G. Galstyan. 2019. Auto-encoding total correlation explanation. In *AISTATS*.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *ICLR*.

Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. 2022. AvatarCLIP: Zero-shot text-driven generation and animation of 3d avatars. *arXiv preprint arXiv:2205.08535*.

Ajay Jain, Ben Mildenhall, Jonathan T Barron, Pieter Abbeel, and Ben Poole. 2022. Zero-shot text-guided object generation with dream fields. In *CVPR*.

Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg L. 2014. ReferItGame: Referring to objects in photographs of natural scenes. In *EMNLP*.

Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In *ICML*.

Juil Koo, Ian Huang, Panos Achlioptas, Leonidas J Guibas, and Minhyuk Sung. 2022. Partglot: Learning shape part segmentation from language reference games. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16505–16514.

Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. 2018. Variational inference of disentangled latent concepts from unlabeled observations. In *ICLR*.

David Lewis. 1969. *Convention: A philosophical study*. Harvard University Press.

Minghua Liu, Minhyuk Sung, Radomir Mech, and Hao Su. 2021. Deepmetahandles: Learning deformation meta-handles of 3d meshes with biharmonic coordinates. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12–21.

Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. 2022. Text2Mesh: Text-driven neural stylization for meshes. In *CVPR*.

Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy J Mitra, and Leonidas J Guibas. 2020. StructEdit: Learning structural shape variations. In *CVPR*.

Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. 2019. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *CVPR*.

Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. 2021. StyleCLIP: Text-driven manipulation of stylegan imagery. In *ICCV*.

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

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *ICML*.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. 2022a. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. 2022b. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*.

Minhyuk Sung, Zhenyu Jiang, Panos Achlioptas, Niloy J. Mitra, and Leonidas J. Guibas. 2020. DeformSyncNet: Deformation transfer via synchronized shape deformation spaces. *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia)*.

Jesse Thomason, Mohit Shridhar, Yonatan Bisk, Chris Paxton, and Luke Zettlemoyer. 2021. Language grounding with 3d objects. In *CoRL*.

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

Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. 2022. CLIP-NeRF: Text-and-image driven manipulation of neural radiance fields. In *CVPR*.

A Appendix

In this appendix we will provide further detail on the implementation of our models, and the parameters during training and editing. We will also provide the mathematical definition of Part-wise Edit Precision (PEP), and the specific implementation of Output-Driven Edit Step-Size Adjustment (ODESSA). Through ablations, we will demonstrate the importance of language disentanglement for the task of localized shape edits, and also the importance of our edit-time design decisions (namely, ODESSA and Neighborhood Simplex Editing). Finally, we'll provide further details on the human evaluation conducted to evaluate our method.

A.1 Implementation details

When learning the joint space, we train using batch-sizes of 64, with 256-dimensional encodings for every shape (for both ShapeGF and IM-NET autoencoders). We pretrain the shape autoencoders and the joint spaces on the official ShapeTalk training set. For both IM-NET and ShapeGF, we use the default training hyperparameters, matching the pretraining process of autoencoders in ChangeIt3D. For both autoencoders, the latent space is 256 dimensional.

The dimensionality of our model's joint space is 128 dimensional, and use 14 expert networks, each with 3 feedforward layers with skip connections. For the implementation of the transformer, we use a slightly modified version of the text encoder used for CLIP with 4 layers and 4 multi-attention heads, which produces a sentence embedding that is 1024 dimensional at the first token.

The voting network is implemented as a single fully connected layer, with the associated softmax temperature parameter (for producing weights of experts) initialized at 1.0 during training. We set the relative weight between the binary classification loss and LADIS loss to be 1 – that is, our training loss is $\mathcal{L} = \mathcal{L}_{binary} + \mathcal{L}_{LADIS}$. Training is done for 20 epochs and the checkpoint with the highest validation accuracy on the official ShapeTalk validation set is selected.

When editing using the pretrained joint space, the number of neighbors we choose for NSE is 64. We also use gradient descent for 50 steps, with each step rescaled according to ODESSA.

Step	Multiutt LADIS	ShareCon LADIS	Random LADIS	w/o LADIS
1	**0.372**	0.345	0.271	0.185
2	**0.384**	0.320	0.217	0.191
3	**0.373**	0.316	0.252	0.182
4	**0.321**	0.303	0.252	0.171
5	**0.305**	0.269	0.241	0.162

Table 4: Different variants of LADIS, from most ideal to least, from left to right. mPEP shown for 5 edit steps, where the output of the previous step is the input shape into the next step, for the same edit description. We see that mPEP generally decreases from the most to least ideal variant of LADIS.

A.2 Definition of Part-Wise Edit Precision

Part-Wise Edit Precision is a novel metric we introduce to measure the locality of an edit to the parts that are explicitly mentioned by language instructions. The metric measures the log of the ratio between percentage volume change found in the correct regions of the input 3D shape versus the entire shape.

Let sequence of edits to a shape be $s(t)$, where $s(0)$ is the original input shape. We assume that for input shape $s_0 = s(0)$, we have a part decomposition $R_{s_0} : \mathbb{R}^3 \rightarrow P$ that maps a given point in $\mathbb{R}^3$ to a label within P, the full set of part labels. We can construct R_{s_0} by using part-segmentation masks from PartNet.

We also construct a mapping $Q : L \rightarrow \mathcal{P}(P)$, which is a manually designed classifier that, based on the words within the utterance u from space L, selects a subset of P as "relevant" parts, where changes should be allowed. We define Q manually by mapping sets of words found in utterances of ShapeTalk to different PartNet part classes.

Given an implicit occupancy decoder *dec*, we can find the volume difference Δv defined for some edit $s' = s(t)$, and some region X:

$$\Delta v(X, s') = \left| \int_{x \in X} dec(s', x) - dec(s_0, x) dx \right| \tag{4}$$

We can then find the percentage volume change in region X as:

$$w(X, s') = \frac{\Delta v(X, s')}{\int_{x \in X} dec(s_0, x) dx}. \tag{5}$$

PEP is defined as the log of the ratio of percentage volume change found in the parts of the input object mentioned by language to the percentage volume change found in the whole object:

$$PEP(s(t), u) = \log \frac{w(\mathcal{X}_{s(0),u}, s(t))}{w(\mathcal{X}, s(t))}, \tag{6}$$

where $\mathcal{X}$ is the ambient space and $\mathcal{X}_{s(0),u} = \mathcal{X} \cap R^{-1}_{s(0)}(Q(u))$ is the set of points that belong in the set of parts mentioned by utterance u. When an edit is more "local" in a part-wise sense, we expect PEP to be high. Since the ratio is established between two percentage changes, this effectively normalizes against the size of the part mentioned, which rewards nominally smaller volume changes to skinny/small parts mentioned by language if it amounts to a significant percentage of the original part volume. We find empirically that this is important, as many utterances refer to volumetrically less-significant parts, like legs and armrests.

We've also found empirically that the $\log$ of the ratios has a much more Gaussian-like distribution, which matches the human prior we assume over visual obviousness of a change in the correct parts.

Due to the use of part segmentation masks (used to define R_{s_0}), PEP is susceptible to errors in the masks, as well as translations and rotations within edits that may shift parts far outside the groundtruth masks. The latter may be mitigated by measuring and comparing edits of smaller magnitude, or "swelling" the groundtruth masks, as we have done in our implementation. Additionally, the PEP metric depends on a predefined set of part semantic groups (e.g. for chairs, we've used "back", "seat", "legs" and "armrests"), and is therefore specific to a fixed part semantic group for each object category.

However, since PEP does not have to rely on a pre-trained part classifier (as done in the metric proposed by ChangeIt3D) and accounts for the differences in size of parts by assigning volumetrically smaller parts higher weight, we believe that it is still a stronger metric than what currently exists. PEP therefore should be used to do system-level comparisons of edit locality.

To represent the size of the edit, we measure the mean volume change mΔv at edit timestep t as the mean of $v(\mathcal{X}, s(t))$ over different (source shape $s_i(0)$, language u) pairs. Over the same set, we measure the mean PEP at a certain edit timestep t as the mean over all edits:

$$mPEP = \frac{1}{N}\sum_{i=1}^{N} PEP(s_i(t), u_i) \qquad (7)$$

Note that while the above is formulated for implicit occupancy autoencoders like IM-NET, it can be extended to other types of autoencoders by other notions of edit distance, such as chamfer distance between surfaces.

	Multiutt	w/o ODESSA	w/o NSE
mΔV	0.0334	0.0095	0.0274
FID	33.58	21.36	46.04

Table 5: Ablations in different ablations in the edit-time procedure. We verify through this that removing the ODESSA step results in less obvious volumetric changes, and that without NSE, our method produces less plausible edit outputs.

A.3 Implementing ODESSA

ODESSA rescales the size of each step to combat the problem that latent space traversal often produces unpredictable differences in the space, which can be especially detrimental when we optimize within the joint space over multiple optimization steps.

Thus, ODESSA rescales the size of each step according to some desired degree of total change in the decoded output per optimization step, denoted by the parameter δ:

$$ODESSA(s', \Delta\epsilon, \delta) = \delta \times \left| \left(\nabla_s \int_{x \in \mathbb{R}^3} \text{dec}(s, x) dx \right)^T (\Delta\epsilon^T Q) \right|^{-1}$$

Note that in the above formulation, we assume that the geometry is outputted by the decoder as an implicit representation, which means allows the integral over the 3D space to capture total variation within the output occupancy values. But as is the case with PEP, the following formulation can also be extended to other autoencoder architectures by using other notions of edit distance like chamfer distance between surfaces.

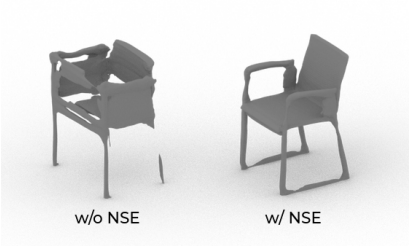


Figure 8: An example of the same optimization-based editing, run with and without NSE. Note how without NSE, the optimization produces spurious artifacts that degenerates the chair output.

A.4 Ablations

In Table 4, we compare the effect of gradually relaxing the assumption of language independence on the improvement of the edit locality across multiple edits. We do these experiments on the ShapeTalk dataset, which supports the strongest assumptions of edit description independence (i.e. multiutterance LADIS). We find that the better the negative set is (i.e. stronger independence), the higher the mPEP performance. This suggests that the degree to which we can mine for independent utterances heavily influences the degree of edit locality, and furthermore that this trend holds throughout iterative edits.

In Table 5, we compare the effects of removing NSE and ODESSA. Specifically, since these two techniques are meant to preserve the shape class and also produce more obvious edits to shapes, we compare the Frechet Inception Distance (FID) of the renderings of decoded edits to that of the distribution of the decoded input latent codes. We find that without using NSE, our model's FID largely increases, while we notice a slight decrease in the mean amount of occupancy difference in the output shape. This suggests that without using NSE, edits easily shift the edited shape off of the valid shape manifold (See Figure 8). On the other hand, without using ODESSA to recalibrate the optimization step sizes, the mean amount of occupancy difference (mΔV) becomes comparable to the ChangeIt3D baseline (see results in Table 1) – that is, the edits become measurably less obvious.

These ablations therefore suggest that our two modifications to the canonical optimization-based editing procedure both allow for more obvious edits while improving the ability of our edits to stay on the valid shape manifold.

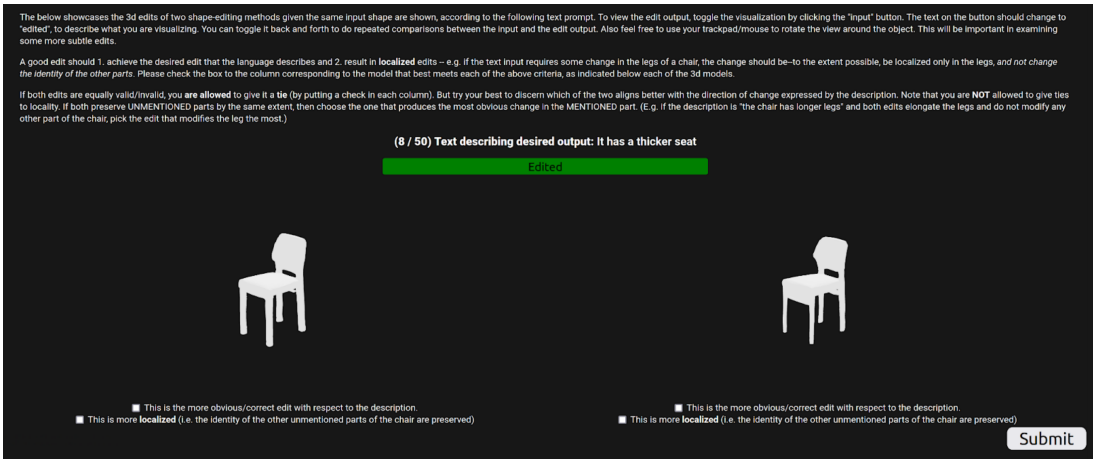


Figure 9: The web interface used by human evaluators to annotate the quality of shape edits. We display the two edits side by side, one of which is produced by ChangeIt3D, and the other from our method. Through repeatedly clicking the button in green, we allow users to toggle the view from inspecting the *input* shape to the edited *output*, so that minute differences in edits (e.g., slight scaling of certain parts) of both methods can be more easily detected than side-by-side comparisons. Users can also rotate the 3D models around, as well as zoom in/out, to inspect the influence of the edit on different parts of the input. Before moving onto the next sample, the evaluator must select which edit is more semantically obvious/correct, and which is more localized. We clearly define what each of these criteria mean in the instructions.

A.5 Human Evaluation

Figure 9 shows the interface we used to collect human evaluation data to evaluate our qualitative results. We asked 10 human evaluators to each annotate 50 sample edits (500 annotations total), comparing our method and ChangeIt3D. 8 of the 10 annotators have no experience with NLP, graphics or machine learning. No time limit was enforced on the annotation process. The average time spent on annotations was 35 minutes.

Each evaluator is given a 2 minute introduction to the task, primarily explaining how to use the interface, and defining and providing examples for "edit locality" and "edit correctness". Importantly, evaluators are asked to comparing edits in *both* criteria.

Because some of the 3D edits are fairly subtle, the "toggling functionality" enables the user to quickly flip back and forth between the source and target shapes. We found this to be very useful in deciding edit locality and correctness of relatively smaller or thinner parts (e.g. legs of chairs.).

We also found that for structural changes (e.g. "the chair has stretchers"), human evaluators typically have a harder time deciding which one is more obvious or accurate if both (or neither) edits being compared achieve the structural change. As such, we allow for ties in the correctness/obviousness criterion.