Unpaired Translation of 3D Point Clouds with Multi-part Shape Representation

CHIH-CHIA LI, National Yang Ming Chiao Tung University, Taiwan
I-CHEN LIN*, National Yang Ming Chiao Tung University, Taiwan

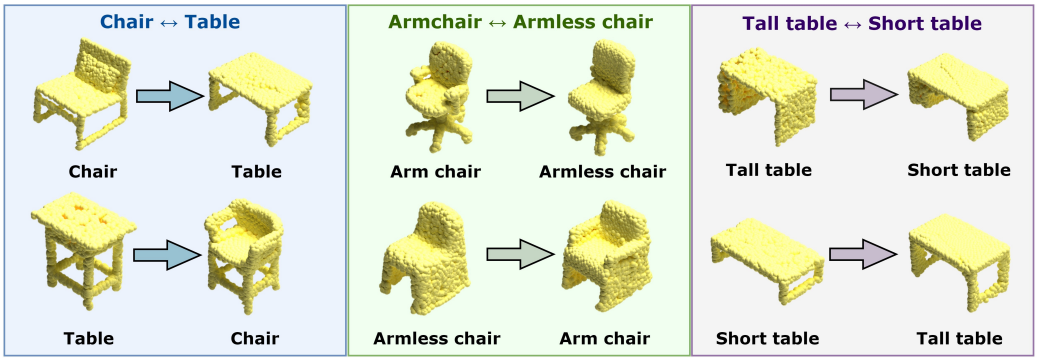

Fig. 1. Cross-domain shape translation by the proposed framework. Given two sets of point-cloud data, our framework learns the translation between these two domains with multi-part representation, in which more local details, such as the table or chair leg structure, can be preserved during translation.

Unpaired shape translation is an emerging task for intelligent shape modelling and editing. Recent methods for 3D shape transfer use single- or multi-scale latent codes but a single generator to generate the whole shape. The transferred shapes are prone to lose control of local details. To tackle the issue, we propose a parts-to-whole framework that employs multi-part shape representation to preserve structural details during translation. We decompose the whole shape feature into multiple part features in the latent space. These part features are then processed by individual generators respectively and transformed to point clouds. We constrain the local features of parts within the loss functions, which enable the model to generate more similar shape characteristics to the source input. Furthermore, we propose a part aggregation module that improves the performance when combining multiple point clusters to generate the final output. The experiments demonstrate that our multi-part shape representation can retain more shape characteristics compared to previous approaches.

CCS Concepts: • **Computing methodologies** → **Shape modeling**; *Unsupervised learning*.

Additional Key Words and Phrases: Unpaired domain translation, generative adversarial network, multi-part shape modeling, shape deformation

*I.-C. Lin is the corresponding author.

Authors' addresses: Chih-Chia Li, nycu.309553007.cs09@nycu.edu.tw, National Yang Ming Chiao Tung University, Hsinchu City, Taiwan; I-Chen Lin, ichenlin@cs.nycu.edu.tw, National Yang Ming Chiao Tung University, Hsinchu City, Taiwan.

2577-6193/2023/5-ART $15.00
https://doi.org/10.1145/3585508

ACM Reference Format:
Chih-Chia Li and I-Chen Lin. 2023. Unpaired Translation of 3D Point Clouds with Multi-part Shape Representation. *Proc. ACM Comput. Graph. Interact. Tech.* 6, 1 (May 2023), 20 pages. https://doi.org/10.1145/3585508

1 INTRODUCTION

In recent years, unpaired image-to-image translation has gained a great deal of interest in computer graphics, vision, and machine learning fields. Due to the inconvenience of collecting pairwise information, most methods aim to learn translation between two domains without pairwise supervision. Besides, they typically focus on transferring the style in an image, such as color or texture. For shape translation, the goal is to generate an output shape for a different domain while preserving certain features from the input, and it has various applications such as 3D model generation, 3D asset extension, furniture design assistance, and so forth. To date, there are relatively fewer research works on shape-to-shape translation. P2P-NET [Yin et al. 2018] develops a general-purpose neural network to learn geometric transformations between point sets from two domains, e.g., meso-skeletons and surfaces, partial and complete scans. Despite successfully changing shape, P2P-NET requires paired training data. VC-GAN [Gao et al. 2018] proposes an automatic unpaired deformation transfer. Although pairwise correspondences are not required, it can only achieve minor shape deformations without significant structural variation, e.g., a fit person to a fat person, or changes in posture.

Our work aims at translating shapes across domains with essential structural variations and keeping the distinct shape characteristics. For example, when a chair is converted to a table, the shape of table legs looks similar to the shape of original chair legs. LOGAN [Yin et al. 2019] and UNIST [Chen et al. 2022] are the most relevant methods to our work. LOGAN can perform both content and style transfers between shapes. However, the translation network uses a single autoencoder architecture to first encode a whole input shape into the multi-scale latent codes that represent global features. Although the latent codes concatenate multi-scale shape features, this operation can still omit distinctively regional information, making it difficult to keep local details during shape translation. To address this issue, UNIST combines the merits of both latent-space processing and position awareness and proposes a latent grid representation. It enables large shape transforms and well preserves fine local details for shape translation. UNIST uses neural implicit functions to represent 3D shapes. By contrast, point clouds are more friendly for shape editing, since they explicitly express the shapes and are efficient in computation and memory usage. This urged us to investigate an effective representation to achieve unpaired shape translation based on point clouds and preserve the local features from the source shape.

Recently, several deep learning methods propose using local-to-global frameworks to better construct local details [Chen et al. 2021, 2020; Gao et al. 2019]. They decompose an input image or object into multiple small parts and demonstrate that modeling the shape space as low-dimensional components can improve the quality of local detail synthesis. Furthermore, in 3D reconstruction or generation, several approaches reconstruct point clouds with multiple simple primitives or point clusters to improves the quality of 3D objects [Deprelle et al. 2019; Groueix et al. 2018; Sun et al. 2021; Zhang et al. 2022; Zhao et al. 2019]. However, unlike human face images where semantic features (i.e., eyes, nose and mouth) can be applied for matching or alignment [Lee et al. 2018], it is not easy to extract meaningful regions from an arbitrary point cloud. Without additional segmentation information, we adopt decomposing the point cloud into part features in the latent space instead of the original 3D space.

Inspired by the above, we propose a neural network based on the parts-to-whole strategy for the task of unpaired shape-to-shape translation. The parts-to-whole strategy means that we first generate multiple point clusters separately, and then combine these point clusters to form a whole

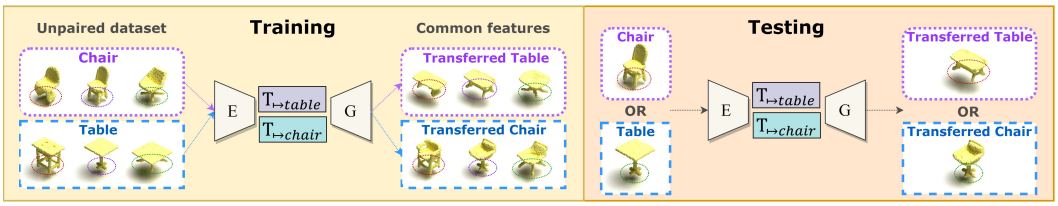

Fig. 2. The concept of the training and testing process. In the training process, our input is unpaired point cloud datasets from two domains. The transferred results generated by our network are expected to have common shape features with the input. In the testing process, given an input point cloud from one domain, our framework generates an output with similar shape characteristics in the other domain.

point cloud. As shown in Figure 1, we can translate a chair to a table which has similar shape characteristics (e.g. the legs of the chair and table). We also converted an armchair to its armless counterpart and a table from tall to short domains, while retaining its originally main structure.

Figure 2 shows the concept of our training and testing processes. We train our model using unpaired data from two domains. Taking Chair ↔ Table translation as an example, our training dataset has both unpaired chairs and tables. The proposed network has to implicitly learn the common features existing in two domains. Dotted circles with the same color between the input chairs (tables) and output tables (chairs) indicate parts with similar shape characteristics that we intend to retain. In the testing process, we can send any chair or table to our model to generate the corresponding transferred result. Different from prior work, we do not use a single generator to generate the entire shape. As shown in the overview Figure 3, we employ independent feature mapping modules to extract multiple local part features. These part features are then sent to the point generators through individual branches, respectively. The point generator can transform latent features to point clusters.

However, since each point cluster is independent of the other, directly aggregating multiple point clusters cannot achieve satisfactory results. Therefore, we propose a part aggregation module, which introduces global information into each part and deforms each point cluster. Moving 3D points of each part according to the estimated displacement can further fit the generated result to the point cloud reconstructed by autoencoder. Besides, our model can automatically segment the point cloud without supervised data. We observed that translating shapes by parts reduces the difficulty for the network to model new shapes and enables the preservation of fine geometric details (e.g. crossbars of chairs and tables) during shape translation.

Through extensive qualitative and quantitative experiments, we demonstrate the advance of our approach for unpaired shape-to-shape translation. Compared with related methods, our translation results more accurately preserves distinctive local details, making it more similar to the shape characteristics of the input source. We also conducted ablation studies to verify the effect of part decomposition and local part deformation. In summary, the main contributions of our work are as follows:

- We present a novel parts-to-whole framework for unpaired shape-to-shape translation which preserves more distinctive structural details from the source.
- The proposed part aggregation module improves the performance when combining multiple point clusters by incorporating global feature of shape and deforming local point sets.
- Experiment results shows the superiority of the proposed approach in both qualitative and quantitative evaluations compared to related state-of-the-art approaches.

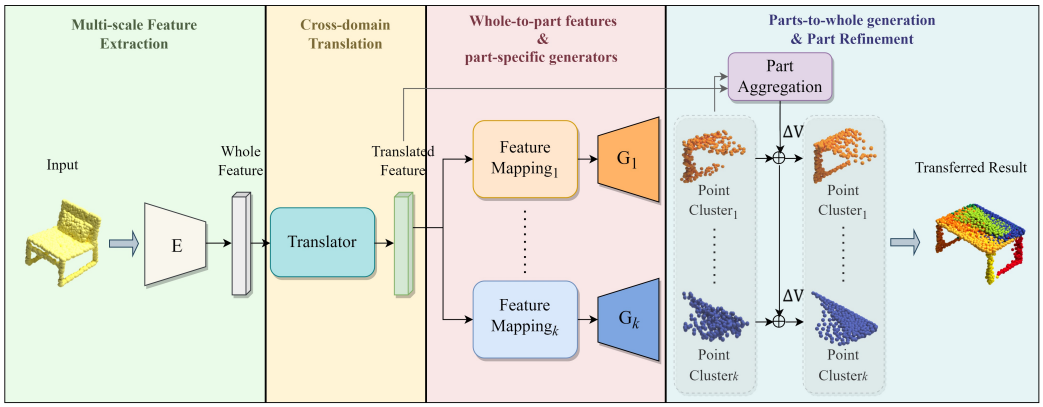

Fig. 3. Overview of the proposed parts-to-whole framework. Multiple part-specific generators are employed for translation by parts, and these parts are then gathered and refined by a part aggregation module.

2 RELATED WORK

2.1 Point Clouds with Deep Networks

Since point clouds directly represent the geometric coordinates and are relatively easy to edit, they have been widely used in 3D shapes modeling. To process such sparse and unordered data, several methods have been proposed to process point clouds with permutation-equivariant networks. A key breakthrough in connecting point clouds to neural networks is PointNet [Qi et al. 2017a]. It uses multi-layer perceptrons and global pooling operations to solve the permutation invariant problem. To concentrate more on local structures, PointNet++ [Qi et al. 2017b] applies PointNet recursively in a hierarchical manner and leverages neighborhoods at multiple scales. The idea of local structures is further developed by DGCNN [Wang et al. 2019b], which proposes the edge convolution over graphs to incorporate local neighborhood information.

However, these methods usually use the max-pooling operation in the encoding phase, fine-grained information is lost and difficult to recover in the decoding phase. To address this issue, PCT [Guo et al. 2021] presented a novel framework based on Transformers [Vaswani et al. 2017]. Since Transformers have a strong capability to handle long-sequence inputs, they can handle the permutation invariant problem of the point cloud. Although the attention mechanism in Transformers is effective in extracting global features, local geometrical information may be ignored. To make Transformers more suitable for point cloud, PoinTr [Yu et al. 2021] introduces a geometry-aware block, which can be plugged into any transformer architecture. With this geometry-aware block, the model can better capture the geometric relation. Motivated by the above, we followed PoinTr and incorporated a geometry-aware transformer mechanism into our encoder. In order to make the representation more thorough, we adapted our feature extractor to be a multi-scale graph-based model.

2.2 Unpaired Domain Translation

Translating an input from the source domain to the target domain without paired data supervision is a challenging task. One of the main difficulties is that there are multiple possible mappings between the two domains, which can lead to instability of training and fails of translation. To tackle this problem, pioneer works [Kim et al. 2020, 2017; Park et al. 2021; Yi et al. 2017; Zhu et al. 2017] proposed the concept of cycle consistency. It assumes that the generated image can be translated

back by inverse mapping and the cycle-reconstructed image should be as similar to the input as possible.

Inspired by these methods, LOGAN uses the cycle consistency loss to train their translation network. In addition, they also added a feature preservation loss, which is formally equivalent to the identity loss [Taigman et al. 2016]. Our translation network is also trained with the same loss set as LOGAN. Unlike LOGAN, which only considers the whole feature, we further evaluate and exploit features of multiple parts. This makes cyclic reconstruction more similar to the input point cloud and facilitates the cross-domain translation.

2.3 3D Part-based Modeling

As the availability of 3D shape datasets with part annotations [Chang et al. 2015; Mo et al. 2019b; Yi et al. 2016] increases, more and more research is interested in 3D part-level object understanding. Several researches promote decomposing 3D shapes into individually meaningful parts using either supervised or unsupervised learning. Previous methods [Delanoy et al. 2018; Niu et al. 2018; Sun et al. 2019; Tulsiani et al. 2017; Zou et al. 2017] learn to assemble objects using cuboid structure and require only a small number of parameters. Approaches such as [Ganapathi-Subramanian et al. 2018; Genova et al. 2019; Kim et al. 2013] utilize the efficiency of template 3D shapes and transfer their structural information onto new objects, while other methods [Li et al. 2017; Mo et al. 2019a; Wu et al. 2020] generate 3D shapes by parts.

Recently, 3DPointCapsNet [Zhao et al. 2019] is based on renowned 2D capsule networks [Sabour et al. 2017] and extends them for 3D data. With multiple independent convolutional layers to capture 3D local features and dynamic routing, 3DPointCapsNet extracts powerful latent representation from input point clouds. In their decoder, they followed the AtlasNet [Groueix et al. 2018] convention. AtlasNet samples 2D points uniformly in the unit square for each latent representation and jointly learns to deform a 2D point set to a surface. Sun et al. [2021] also used the capsule encoder to learn part decomposition. However, they assigned each point to one of the K parts through multiple attention maps and their decoder architecture was similar to AtlasNetV2 [Deprelle et al. 2019]. AltasNetV2 also adopted surface-parametric approaches from AtlasNet, but they used trainable grids called learnable elementary structures. Although these methods performed adequately on point cloud reconstruction, they were usually troubled by outliers and rough surfaces. AXform [Zhang et al. 2022] proposed an attention-based decoder to address these issues and generated point clouds using only latent features. In addition, AXform had fewer network parameters and faster convergence speed. For the above advantages, we adopted AXform as our point generator. Besides, we further present a part aggregation module to improve the multi-part results.

2.4 Point Set Displacement

Displacement-based approaches have been applied to several related work. Yin et al. [2018] implemented the skeleton-to-shape transformation. They fed the learned noise-augmented feature vectors to a set of fully connected layers to produce pointwise 3D displacement vectors. By applying these displacement vectors to the source shape, they could transform a source point set into a target one. Wang et al. [2019a] also proposed a network to estimate per-vertex offsets, which would deform the source model to resemble the target. Unlike other approaches, the offsets in their work were learned according to point locations and global features of the source and target. Wang et al. [2021] introduced a novel 3D representation method that uses the spherical point cloud as the template. They progressively deformed the template by point-wise offsets to fit the target shape in a coarse-to-fine manner.

Although these methods achieve impressive results, they cannot significantly change the shapes. Besides, 3D objects after point-wise displacements still belong to the same domain. In this work,

we propose a part aggregation module to learn per-point displacement. Our module divides a 3D object into multiple smaller point clusters and each point cluster is processed and displaced separately. With such a design, our framework has more opportunity to learn the details during the cross-domain translation.

3 METHOD

The main issue of shape translation is how to well preserve distinctive structure and fine local details. Related approaches use a single encoder and generator to generate the translated shape. Since learning both global and local features with a single generator is difficult, we found that such a strategy is prone to lose local features of the source shape during translation.

To alleviate this problem, we design a novel network that includes a transformer-based encoder and multiple part-specific generators. We use individual feature mapping modules to extract different part features and translate shapes by parts. Meanwhile, our network can learn to automatically segment point clouds into multiple parts. For better parts aggregation, we propose a part aggregation module that incorporates the global feature and deforms local point clusters. Our network and its training procedure are illustrated in Figure 4.

3.1 Overview of our network

Our goal is to translate shapes from the source domain $\mathcal{S}$ to the target domain $\mathcal{T}$. Without additional annotated information, our network has to implicitly learn the common features existing in two domains. In the autoencoder network, as shown in Figure 4a, our encoder E encodes point clouds from two domains into latent space. In latent space, we describe $Z_{\mathcal{S}}^{w}$ and $Z_{\mathcal{T}}^{w}$ as latent codes in domain $\mathcal{S}$ and domain $\mathcal{T}$, respectively, and the superscript w represents the whole shape features for point clouds. We have k feature mapping modules that take the whole latent codes as input to extract multiple local part latent codes $Z_{\mathcal{S}}^{p}$. These part features are then sent to different generators to be converted into point clusters, which are concatenated to form reconstructed coarse output. To better combine point clusters, our part aggregation module introduces the global features into local parts and adds per-point displacement to parts. We take the ensemble of these displaced point clusters as our final reconstructed output.

For translation, inspired by LOGAN [Yin et al. 2019], we perform our transformation in latent space. As shown in Figure 4b, we have two translators $T_{\mathcal{S}\mapsto\mathcal{T}}$ and $T_{\mathcal{T}\mapsto\mathcal{S}}$. The translator $T_{\mathcal{S}\mapsto\mathcal{T}}$ can transfer a latent code from domain $\mathcal{S}$ to domain $\mathcal{T}$, while $T_{\mathcal{T}\mapsto\mathcal{S}}$ performs on the opposite direction. Our training process is different from LOGAN [Yin et al. 2019] and UNIST [Chen et al. 2022], they trained their autoencoder and translators separately. We train our autoencoder and translators alternately. When we train our autoencoder, we fix the parameters of translators, and vice versa. The reason for this training strategy is that if the autoencoder is trained first, the feature distribution will be fixed and thus limiting the learning of the translators. Through such alternate training, our autoencoder and translators can gradually be enhanced.

3.2 Part-level Translation

LOGAN [Yin et al. 2019] used the multi-scale latent codes to represent the entire shape of the input point cloud. The latent codes can be used to recover the shape by a single generator. However, we observed that such a design may lose important local details during translation. Instead, UNIST [Chen et al. 2022] proposes position-aware encoding to incorporate positional information into a latent grid. They employed a neural implicit representation of shapes, rather than point clouds. By contrast, we propose a parts-to-whole model for point clouds, which adopts multiple part-specific generators to generate point clusters and finally integrates them to form a whole point cloud. Our network architecture is introduced as follows.

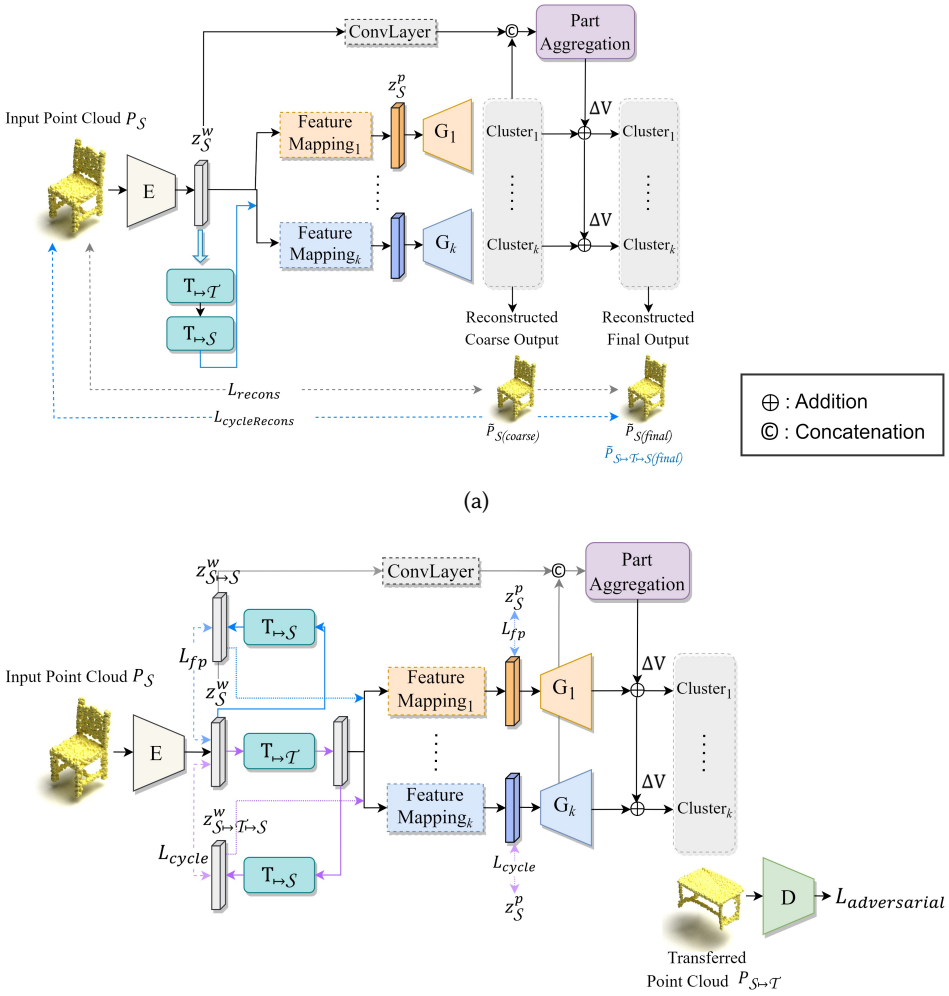

Fig. 4. Overview of training our network. We exploit an alternate training strategy which can be divided into two parts: (a) autoencoder training and (b) translator training. In (a), we use a single encoder and multi-branch feature mapping modules and generators. Our autoencoder encodes shapes from two input domains and generates reconstructed coarse and final outputs. In (b), we have two translators to transfer point clouds between domain $\mathcal{S}$ and domain $\mathcal{T}$. Our translator is trained in an adversarial setting to ensure that the transferred point cloud matches the target domain through the discriminator.

Multi-scale encoder. Our encoder is inspired by PoinTr [Yu et al. 2021], in which a transformer-based encoder is applied to reduce the loss of fine-grained information. They used a lightweight DGCNN [Wang et al. 2019b] as their feature extractor. To get a more comprehensive shape representation, we employ a powerful feature extractor to preserve details. Following the procedure in [Hassani and Haley 2019], we use a series of graph convolution, convolution, and pooling layers in a multi-scale fashion to learn shape features for point clouds Our encoder is illustrated in Figure 5. Deriving the idea of the graph convolution from DGCNN, we use K Nearest Neighbor (KNN)

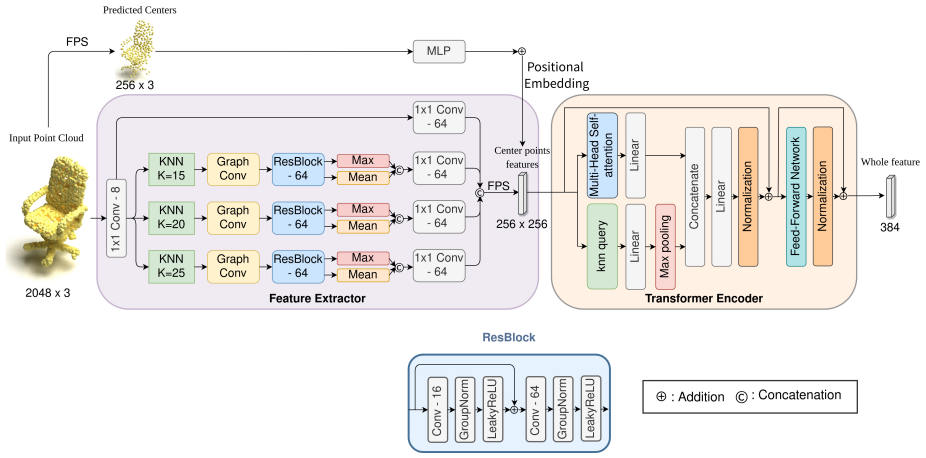

Fig. 5. The architecture of our transformer-based encoder. We adopt a multi-scale graph-based model modified from [Hassani and Haley 2019] as the feature extractor. It takes multi-scale features at each FPS(furthest point sample) center position. The following transformer encoder takes the center features with positional embedding as input, and outputs the fused feature vector representing the whole shape.

with multiple K parameters to form multiple neighbor graphs and extract multi-scale features. We concatenate these multi-scale features to form per-point features. We downsample the points according to centers extracted by the furthest point sample (FPS) approach. The 256-channel features at these 256 extracted point centers can approximate the regional features of the model. Afterwards, the following transformer encoder then fuses these center-point features to be the whole shape features by self-attention, feed-forward and other network mechanisms.

Point generator. To extract the part features, the whole feature is fed into separate feature mapping modules with MLP architecture. In our generator, we take the attention-based generator proposed in AXform [Zhang et al. 2022] as the backbone. Compared to other parts-to-whole generators, AXform generates fewer outliers and network parameters and converges faster. We employ this attention-based generator to convert the part features into point clusters. These point clusters are then aggregated to generate the target point clouds. The detailed architectures of the feature mapping module and point generator are introduced in the supplementary materials.

Translator. Our translators aim to learn which shape features should be transferred during cross-domain translation. The translator network is derived from the translator in LOGAN. We train our translator in the adversarial setting. Unlike LOGAN and UNIST, they applied two discriminators in two domains separately, and both discriminators work in the latent space. We employ an auxiliary discriminator that can simultaneously judge inputs from two different domains. In such a way, we can reduce memory usage. Our point discriminator first extracts a feature vector by PointNet [Qi et al. 2017a] and predicts the likelihoods through two different branches. The detailed architecture of our discriminator is provided in the supplementary materials.

3.3 Point clusters deformations

Although AXform [Zhang et al. 2022] achieves state-of-the-art results among parts-to-whole methods, we found that their approach has several drawbacks. In their work, each small point cluster is generated according to the corresponding part feature, and each branch for part generation

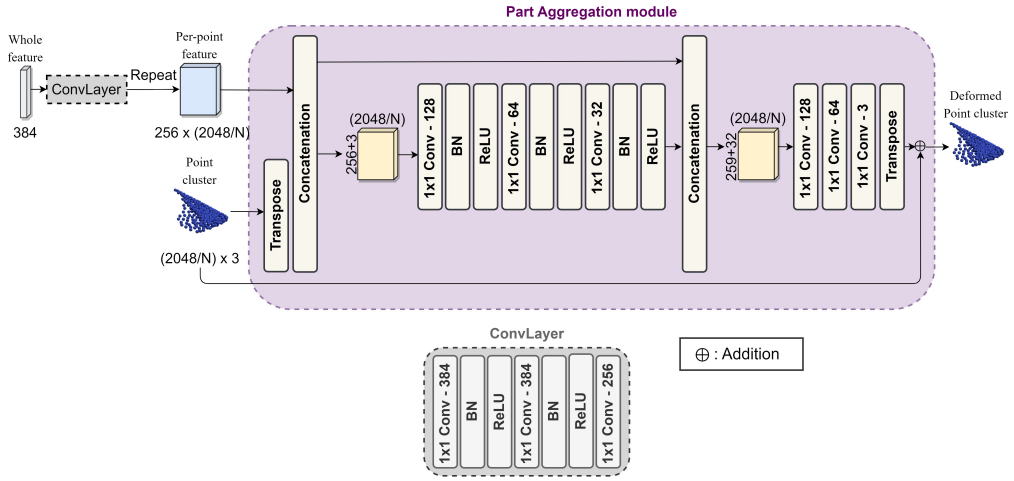

Fig. 6. Architecture of our part aggregation module. The inputs of the part aggregation module are the feature for each point and a point cluster. The part aggregation module can generate a displacement to deform a point cluster. The variable N represents the number of parts.

is independent of the others. When the number of point clusters increases, it is easy to observe gaps or rugged seams among the synthesized point clusters.

To better integrate multiple local parts, we propose a point aggregation module. As shown in Figure 6, our point aggregation module takes the whole feature and one point cluster as inputs and generates per-point displacement for the corresponding local part. The whole feature goes through the convolutional layers to extract important global information for this task before feeding it to the point aggregation module. With our point aggregation module, the global feature of the shape can cooperate with each local feature, which improves the performance when combining multiple point clusters.

3.4 Loss functions

Since we train our autoencoder and translators alternately, the loss functions used in our framework can be divided into two parts: Autoencoding and Translation.

Loss Functions for Autoencoding. In our autoencoder network, we calculate the reconstruction loss not only on the self reconstruction but also on the cyclic reconstruction. The self reconstruction is to directly use the features extracted from encoder to generate point clouds. For cyclic reconstruction, it performs reconstruction with features sent to a translator and inversely-mapped by another translator. To measure the similarity between two point clouds, we adopt Chamfer Distance (CD) [Fan et al. 2017] and Earth Mover's Distance (EMD) [Rubner et al. 2000] for our reconstruction losses.

Given two point clouds P_1 and P_2, the chamfer distance is to calculate the squared distance between each point and its nearest neighbor in the other group set. The chamfer distance (CD) can be defined as:

$$d_{CD}(P_1, P_2) = \sum_{x \in P_1} \min_{y \in P_2} \|x - y\|_2^2 + \sum_{y \in P_2} \min_{x \in P_1} \|x - y\|_2^2 . \quad (1)$$

For measuring point clouds P_1 and P_2 with identical sizes, the EMD distance finds the point correspondences with the minimum distance and reports the distance. It can be described as:

$$d_{EMD}(P_1, P_2) = \min_{\phi:P_1 \to P_2} \sum_{x \in P_1} \|x - \phi(x)\|_2, \tag{2}$$

where $\phi : P_1 \to P_2$ is a bijection.

If we only use the chamfer distance, the generated point cloud is prone to gather in a few regions. We therefore employ both chamfer distance and EMD distance to generate results with more uniform distribution. In the reconstruction loss, we evaluate two kinds of outputs which are the reconstructed coarse output $\widetilde{P}_{\mathcal{S}(coarse)}$ and the final output $\widetilde{P}_{\mathcal{S}(final)}$. The reconstructed coarse output $\widetilde{P}_{\mathcal{S}(coarse)}$ is generated by multiple part-specific generators and the final output $\widetilde{P}_{\mathcal{S}(final)}$ is the result refined by our part aggregation module.

Therefore, our reconstruction loss and cycle-reconstruction loss can be expressed as:

$$\begin{aligned} \mathcal{L}_{recons} =& \lambda_1 d_{CD}(P_{\mathcal{S}}, \widetilde{P}_{\mathcal{S}(coarse)}) + \lambda_1 d_{CD}(P_{\mathcal{S}}, \widetilde{P}_{\mathcal{S}(final)}) \\ &+ d_{EMD}(P_{\mathcal{S}}, \widetilde{P}_{\mathcal{S}(coarse)}) + d_{EMD}(P_{\mathcal{S}}, \widetilde{P}_{\mathcal{S}(final)}), \end{aligned} \tag{3}$$

$$\mathcal{L}_{cycle-recons} = \lambda_2 d_{CD}(P_{\mathcal{S}}, \widetilde{P}_{\mathcal{S}\mapsto\mathcal{T}\mapsto\mathcal{S}(final)}) + d_{EMD}(P_{\mathcal{S}}, \widetilde{P}_{\mathcal{S}\mapsto\mathcal{T}\mapsto\mathcal{S}(final)}), \tag{4}$$

where $P_{\mathcal{S}}$ is input point cloud, $\widetilde{P}_{\mathcal{S}(coarse)}$ is the reconstructed point cloud with coarse shape, $\widetilde{P}_{\mathcal{S}(final)}$ is the reconstructed point cloud with fine shape, and $\widetilde{P}_{\mathcal{S}\mapsto\mathcal{T}\mapsto\mathcal{S}(final)}$ is the point cloud transferred back to its domain. The total loss to train our autoencoder is described as follows:

$$\mathcal{L}_{AE} = \mathcal{L}_{recons} + \lambda_3 \mathcal{L}_{cycle-recons}, \tag{5}$$

and we set the weights λ_1 to 10^4, λ_2 to 10^3, and λ_3 to 0.1

Loss Functions for Translation. In our translation network, we take a similar loss set as LOGAN and further enhance the part similarity. For simplification, we only explain the loss function in details for $T_{\mathcal{S}\mapsto T}$. The loss functions for the opposite direction can be obtained by directly swapping the $\mathcal{S}$ and $\mathcal{T}$ in equations. To achieve cross-domain translation, our discriminator predicts whether the input is real or fake and estimates which class it belongs to. Based on WGAN [Gulrajani et al. 2017], the adversarial loss is formulated as:

$$\mathcal{L}_{WGAN}^{\mathcal{S}\mapsto\mathcal{T}}(P_{\mathcal{T}}, P_{\mathcal{S}\mapsto\mathcal{T}}) = \mathbb{E}[\mathbb{P}(S = real|P_{\mathcal{T}})] - \mathbb{E}[\mathbb{P}(S = fake|P_{\mathcal{S}\mapsto\mathcal{T}})] + \lambda_{gp}\mathcal{L}_{gp}, \tag{6}$$

where $P_{\mathcal{T}}$ is the point cloud in domain $\mathcal{T}$, $P_{\mathcal{S}\mapsto\mathcal{T}}$ is the point cloud translated from domain $\mathcal{S}$ to domain $\mathcal{T}$. $\mathcal{L}_{gp}$ is the gradient penalty proposed by [Gulrajani et al. 2017] for regularization and λ_{gp} is the weight set to 10. For domain classification, we employ cross entropy loss. It is defined as:

$$\mathcal{L}_{class}^{\mathcal{S}\mapsto\mathcal{T}}(P_{\mathcal{T}}, P_{\mathcal{S}\mapsto\mathcal{T}}) = -\mathbb{E}[\mathbb{Y} log\mathbb{P}(C = \mathcal{T}|P_{\mathcal{T}})] - \mathbb{E}[\mathbb{Y}\log\mathbb{P}(C = \mathcal{T}|P_{\mathcal{S}\mapsto\mathcal{T}})], \tag{7}$$

where $\mathbb{Y}$ is the likeihood of real class label.

To encourage one-to-one mapping between the two domains, we apply the cycle-consistency loss to train our translators. We consider both whole shape latent code $Z_{\mathcal{S}}^{w}$ and multi-part latent codes $Z_{\mathcal{S}}^{p}$. The model focuses not only on global features, but also on local features. It makes the latent codes cycle-transferred back more similar to its original ones. Thus, the cycle loss with global and part features is illustrated as:

$$\mathcal{L}_{cycle}^{\mathcal{S}\mapsto\mathcal{T}} = \left\|Z_{\mathcal{S}}^{w} - Z_{\mathcal{S}\mapsto\mathcal{T}\mapsto\mathcal{S}}^{w}\right\|_1^{smooth} + \lambda_{part} \sum_{i=1}^{k} \left\|Z_{\mathcal{S}}^{pi} - Z_{\mathcal{S}\mapsto\mathcal{T}\mapsto\mathcal{S}}^{pi}\right\|_1^{smooth}, \tag{8}$$

where $Z_{\mathcal{S}\mapsto\mathcal{T}\mapsto\mathcal{S}}$ is $T_{\mathcal{T}\mapsto\mathcal{S}}(T_{\mathcal{S}\mapsto\mathcal{T}}(Z_{\mathcal{S}}))$ and k denotes the number of parts in latent code decomposition. λ_{part} is the weight set to 0.1.

LOGAN uses a feature preservation loss which is similar concept to the identity loss in [Zhu et al. 2017]. We also adopt this loss to maintain the invariance of shape characteristics. Similar to our cycle loss, our feature preservation loss also measures both whole and part features to enhance local shape characteristics. It can be formulated as:

$$\mathcal{L}_{fp}^{\mathcal{S}\mapsto\mathcal{T}} = \left\| Z_{\mathcal{S}}^{w} - Z_{\mathcal{S}\mapsto\mathcal{S}}^{w} \right\|_1^{smooth} + \lambda_{part} \sum_{i=1}^{k} \left\| Z_{\mathcal{S}}^{pi} - Z_{\mathcal{S}\mapsto\mathcal{S}}^{pi} \right\|_1^{smooth}, \tag{9}$$

where $Z_{\mathcal{S}\mapsto\mathcal{S}}$ is $T_{\mathcal{T}\mapsto\mathcal{S}}(Z_{\mathcal{S}})$.

In summary, the total loss for training our translators is formulated as follows:

$$\mathcal{L}_{Total} = \mathcal{L}_{WGAN} + \mathcal{L}_{class} + \lambda_{cycle} L_{cycle} + \lambda_{fp} \mathcal{L}_{fp}, \tag{10}$$

where we set λ_{cycle} and λ_{fp} to 20.

4 EXPERIMENTS

4.1 Experimental settings

In our experiments, we applied the ShapeNet Core [Chang et al. 2015] as our dataset which contains more than 50,000 3D mesh models in 55 common categories in total. We trained our model on three pairs of domains, i.e., armchairs and armless chairs, tall tables and short tables, and chairs and tables. In the arm and armless chair translation, we have 2138 armchairs and 3572 armless chairs. 80% of them are used for training and 20% for testing. In the tall and short table translation, we took 2,500 training models and 500 testing models for each domain, respectively. In the chair and table translation, the chair dataset has 4,768 training models and 2,010 testing models, and the table dataset has 5,933 training models and 2,526 testing models. Each mesh model was normalized by setting the longest diagonal of its bounding box to 1, and 2048 points were sampled from each model for our point cloud dataset.

To train our network, we set the batch size to 16 and trained our network for 400 epochs with a single NVIDIA 3090 graphics card. The training time is about 30 hours. we adopted the Adam optimizer with initial learning rates 0.00005, 0.00025, and 0.0002 for our autoencoder, translators network, and discriminator respectively. These learning rates were halved every 100 epochs until 200 epochs. In each iteration, we first trained the discriminator twice and then train the autoencoder and translators once separately.

4.2 Qualitative Comparisons

There has been few methods on unpaired shape-to-shape translation. LOGAN [Yin et al. 2019] and UNIST [Chen et al. 2022] are two impressive approaches aiming at this task. LOGAN uses single generator to recover a point cloud, which may lose control of local details. UNIST addresses this problem by introducing the positional information into latent codes, while it uses volume representation instead of point clouds. To compare these two methods, we reproduced the results by their official codes. Since UNIST employs a neural implicit representation as their 3D shape, we obtained 2048 points from the surfaces of the meshes by the sampling strategy from [Chen and Zhang 2019] for fair comparisons. The comparison of these two methods and ours were conducted on three sets of 3D shape datasets (i.e., armchairs and armless chairs, tall tables and short tables, and chairs and tables).

Chair ↔ Table. The qualitative comparison results are demonstrated in Figure 7. Since LOGAN uses a single generator to construct a point cloud, a few local details are lost. Taking the first row in Figure 7c and the second row in Figure 7g for examples, they are difficult to retain the fine details of the chair and table legs. UNIST incorporates the positional information into latent codes, so

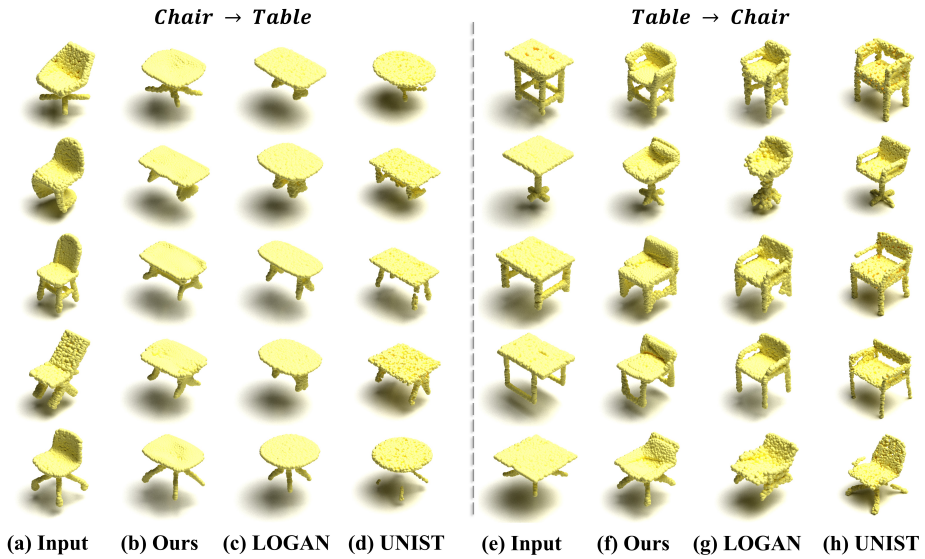

Fig. 7. Comparison of translation results by ours, LOGAN, and UNIST on Chair ↔ Table dataset. (a) Input chairs. (b) Our transferred tables from (a). (c) The transferred tables of LOGAN from (a). (d) The transferred tables of UNIST from (a). (e) Input tables. (f) Our transferred chairs from (e). (g) The transferred chairs of LOGAN from (e). (h) The transferred chairs of UNIST from (e).

they can better preserve local details. However, they can lose control of distinctive input shape characteristics such as the horizontal bars between the chair legs in third and fourth rows of Figure 7e. Our approach is able to preserve the local details and generate results more similar to the shape characteristics of input than other methods on some special shapes (e.g. second row in Figure 7a). We found that sometimes the generated table top is slight rugged due to seams between parts, but it can be smoothed by post-processing methods such as moving least squares (MLS) or applied additional loss for smoothness.

Armchair ↔ Armless chair. Figure 8 illustrates our translation results compared with results by other methods. LOGAN can preserve the general shape of the source input, but the local detailed structures are often dissimilar to the original ones. Taking the second row in Fig. 8c and the first row in Figure 8g as examples, they generated a wrong number of legs compared to the input swivel chairs. Although UNIST preserves fine local details, there are still structural discontinuities in several cases (e.g. first and third rows in Figure 8h). Our method produces the results with structural features more similar to the input, as shown in the first row of *Armchair → Armless chair* translation, only our method can preserve the horizontal bars between the legs of the input chair.

Tall table ↔ Short table. In Figure 9, we show our translation results on tall and short table dataset. It is evident that our method can generate more similar shape characteristics than LOGAN. As shown in the second row in Figure 9g and the fourth row in Figure 9c, LOGAN can only generate part of the horizontal bars between the legs of the input table. UNIST preserves distinctive structure and fine details well, and has high-quality results due to neural implicit shapes. However, in some special shapes, taking the first and third rows of the right hand side of Figure 9, the shape of our translation results is more similar to the source input.

Comparison to retrieval results. In the translation task, we have to verify that our framework indeed learns shape transformations, rather than simply retrieves a training sample from the target

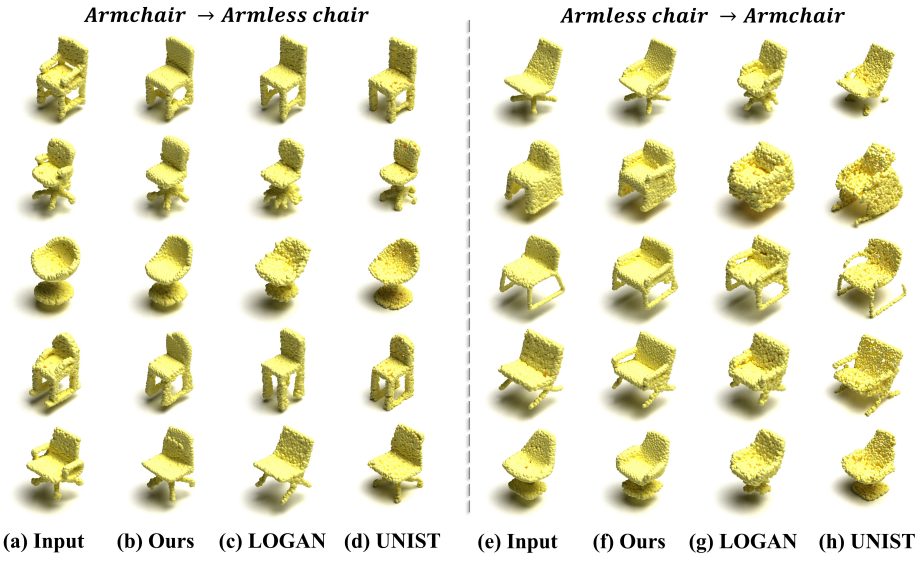

Fig. 8. Comparison of translation results by ours, LOGAN, and UNIST on Armchair ↔ Armless chair dataset. (a) Input armchairs. (b) Our transferred armless chairs from (a). (c) The transferred armless chairs of LOGAN from (a). (d) The transferred armless chairs of UNIST from (a). (e) Input armless chairs. (f) Our transferred armchairs from (e). (g) The transferred armchairs of LOGAN from (e). (h) The transferred armchairs of UNIST from (e).

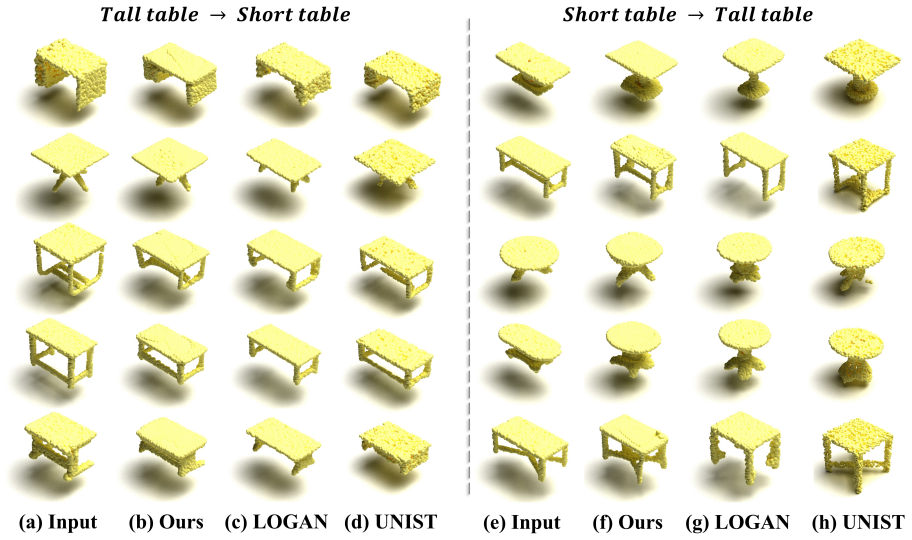

Fig. 9. Comparison of translation results by ours, LOGAN, and UNIST on Tall table ↔ Short table dataset. (a) Input tall tables. (b) Our transferred short tables from (a). (c) The transferred short tables of LOGAN from (a). (d) The transferred short tables of UNIST from (a). (e) Input short tables. (f) Our transferred tall tables from (e). (g) The transferred tall tables of LOGAN from (e). (h) The transferred tall tables of UNIST from (e).

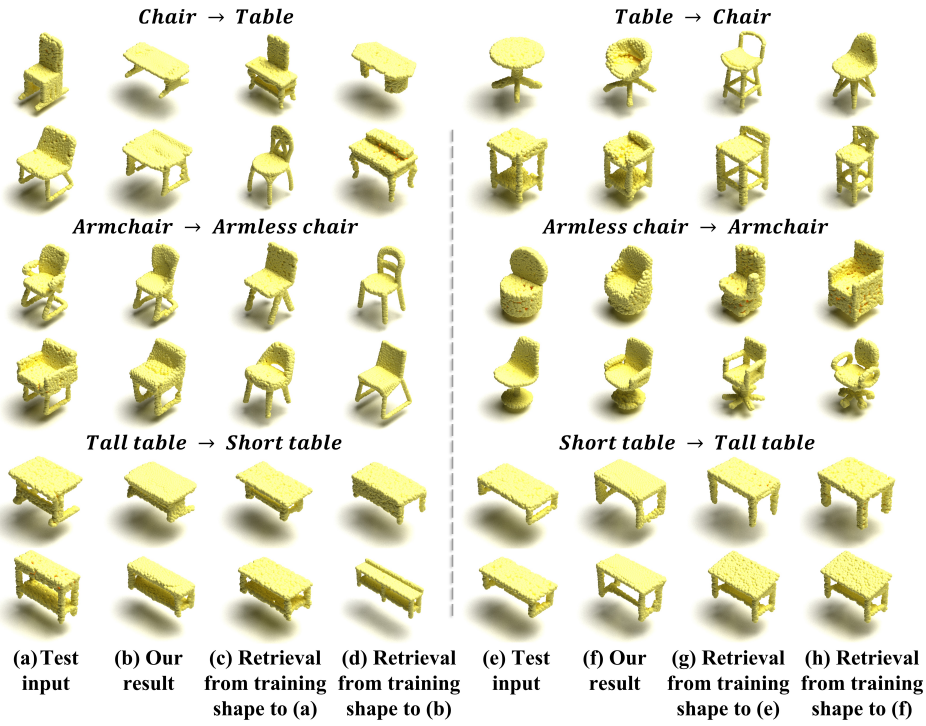

Fig. 10. Qualitative comparison of our translation results and retrieval results on different 3D shape datasets. We use EMD distance to retrieve training shapes from the target domain that are closest to test input and our output. Our translated shapes are novel and not simply retrieved from training sets.

domain. We took the EMD Distance to find training shapes from the target domain that are closest to the test input and to our transferred output. As shown in Figure 10, we demonstrate the retrieval results on different datasets. In Figure 10b-c and Fig. 10f-g, the shape of our translation results is more similar to the test input than those retrieved directly from the training dataset. Besides, our results are quite different from these retrieved training shapes (i.e., Figure 10d and Figure 10h).

4.3 Quantitative Comparisons

Unpaired shape translation does not have a ground truth dataset and translation results can be highly varied. It is difficult to objectively assess the quality of a translated result. In the quantitative comparisons, since there are no suitable metrics or exact ground truth for the chair and table translation, we can only evaluate on other two datasets. In UNIST [Chen et al. 2022], they proposed a way to evaluate 3D translations on Armchair ↔ Armless chair dataset. When converting an armchair and an armless chair, we would expect the transferred chair to have only added or removed armrests, and no other shape changes. Therefore, UNIST used *one-sided Chamfer Distance* (one-sided CD) to evaluate the quality of a translation. Regardless of the direction of the translation, this special CD is calculated from the armless chair to its corresponding chair with armrests. The reason behind is that since the armless chair can be regarded as a part of its corresponding armchair, we can find the distance between each point in the armless chair and its nearest neighbor in the corresponding armchair. As the one-sided CD value is lower, the shape of the translated armless

	Arm→Armless	Armless→Arm
LOGAN†	0.0249	0.0273
UNIST†	0.0234	0.0235
Ours*	**0.0156**	**0.0111**

Table 1. Quantitative comparison with other methods on Armchair ↔ Armless chair dataset using one-sided CD. The best values are highlighted in bold. †: The reported distances of LOGAN and UNIST here are directly extracted from [Chen et al. 2022]. *: The reported distances of ours here are scaled based on LOGAN's distances to fit the scaled distances reported in [Chen et al. 2022].

	Tall→Short		Short→Tall	
	CD	EMD	CD	EMD
LOGAN	0.0021	0.0457	0.0019	0.0431
Ours	**0.0016**	**0.0375**	**0.0018**	**0.0429**

Table 2. Quantitative comparison with LOGAN on Tall table ↔ Short table dataset using CD and EMD distance. The best values are highlighted in bold.

	Testing time (s)	Memory usage (MB)
LOGAN	0.15	8763
UNIST	14.78	1649
Ours	0.14	1883

Table 3. Testing time is the translation time for each point cloud, and its value is in seconds (s). Memory usage is the GPU capacity required by the network during the testing, and the unit is megabytes (MB).

chair is closer to the armless part of its corresponding armchair. However, in the calculation of CD, it is possible that multiple points find the same nearest neighbor, which may bias the measure of the quality of a translation.

We also used this metric to measure our shape translation between chairs with and without armrests. The results are reported in Table 1. UNIST performs on a scaled volume, and they reports the accuracy of their results and the results of LOGAN in that scale instead of the original scale for ShapeNet models. We rescaled the values on our result to fit the scale of UNIST. The scale is evaluated according to ratio between the one-side CD distance of LOGAN in ShapeNet and the distance in volume scale (reported in [Chen et al. 2022]). Our method outperforms the state-of-the-art methods on this metric. It manifests that our model learning both the global and local features benefits the shape translation.

On Tall table ↔ Short table dataset, since the transferred table is only different in height with respect to the input and it should retain the shape characteristics of the input table, we therefore use the scaled table as the ground truth to evaluate the shape similarity between the two tables. We adopted Chamfer distance and EMD distance to measure the similarity between the ground truth produced by us and the transferred result. Table 2 shows quantitative results. Our method can achieve lower CD and EMD distance values than LOGAN. That reveals that our results are closer to the ground truth.

Furthermore, we measured the testing time and memory usage required for our method and other methods to translate a point cloud. As shown in Table 3, our testing time is close to the time used by LOGAN, which is sufficient for applications of interactive shape translation. However, since UNIST uses a neural implicit representation as their 3D shape, they need more time to process

	Arm→Armless	Armless→Arm
w/o part loss	7.69	7.74
w part loss	6.28	4.48

Table 4. The performance of our model with and without part loss is measured by one-sided CD. The part loss is to gauge part features within the cycle-consistency loss and feature preservation loss. The reported one-sided CD has been multiplied by 10^4 for clear comparison.

Point clusters	Arm→Armless	Armless→Arm
1	7.69	7.32
4	7.31	5.17
8	6.88	5.17
16	6.28	4.48
32	5.60	4.63

Table 5. Ablation studies for the number of point clusters that compose of a point cloud by one-sided CD on armchair and armless chair dataset. The reported one-sided CD has been multiplied by 10^4 for clear comparison.

	Arm→Armless	Armless→Arm
w/o PA module	6.79	5.06
w/ PA module	6.28	4.48

Table 6. Ablation studies for our method with and without point aggregation (PA) module by one-sided CD on the armchair and armless chair dataset. The reported one-sided CD has been multiplied by 10^4 for clear comparison.

the data. In the memory usage, UNIST converts the point cloud into latent codes in advance so that the encoder does not need to be loaded during the testing, so the required memory capacity is less than that of LOGAN. The memory usage required by our network is between the two methods and closer to the amount of UNIST.

4.4 Ablation Study

To verify the effectiveness of each component of our proposed method, in the following paragraphs, we discuss the results of loss functions, hyperparameters, and our architecture under different settings.

Effectiveness of part loss. We considered both global and local features on the cycle-consistency loss and feature preservation loss (i.e., the second term in Equation (8) and (9)). To verify that the calculated part loss preserves the local geometric characteristics, we qualitatively and quantitatively analyzed the results. In Tabel 4, the *one-sided Chamfer Distance* value of the method with part loss is lower than that without part loss. This implies that considering the losses on parts allows the model to learn the local shape structures, which makes the transferred chair structure (omitted the armrest) more similar to the input. We also demonstrate the qualitative results in the supplementary materials.

Effectiveness of multi-part shape representation. To verify the efficacy of our multi-part shape representation in preserving local details, we conducted an ablation study on the number of point clusters. The number of point clusters represents how many point groups a point cloud consists of. The results given in Table 5 demonstrate that dividing shape features into more parts will lead

	Arm→Armless	Armless→Arm
(a) Transformer encoder w/ one-scale feature extractor	6.45	5.06
(b) Only multi-scale feature extractor	6.41	4.52
(c) Each part w/ an individual PA module	6.97	5.71
(d) Global feature w/o ConvLayer	6.50	4.71
(e) Whole shape deformed by multiple PA modules	8.72	5.97
(f) Ours	**6.28**	**4.48**

Table 7. Quantitative comparison of different architectures on armchair and armless chair dataset by one-sided CD. The reported one-sided CD has been multiplied by 10^4 for clear comparison.

to higher accuracy. However, when the number is larger than 16, the CD value of *Armless* → *Arm* increases, although there is a decrease in *Arm* → *Armless*. In AXform [Zhang et al. 2022], they show the improvement of performance decreases a lot when the number of point clusters is larger than 16. Therefore, we also chose 16 for the comparison experiments.

Effectiveness of part aggregation module. As mentioned before, combining multiple point clusters generated from independent part features is prone to shape discontinuities. We propose a point aggregation module to make each part aware of the global information and deform them to further fit the generated output to the target point cloud. As shown in Table 6, our method with point aggregation module achieves better performance on one-sided CD. We also demonstrate the qualitative comparisons in the supplementary materials to show the benefits of our part aggregation module.

Comparisons of different architecture settings. To verify the effectiveness of our model design, we show the performance of different architectures in Table 7. Table 7(a) is to replace our multi-scale feature extractor with a one-scale feature extractor using only one KNN proposed by PoinTr [Yu et al. 2021]. Table 7(b) is to remove the transformer and only use the feature extractor with a max-pooling operation to extract the features of the point cloud. Since the multi-scale architecture can represent more thorough features, and the transformer can fuse features, our performance is better than these two settings.

For the aggregation of multiple point clusters, we tried a few different settings. We set the number of point aggregation modules to be the same as the number of point clusters, and fed the same global feature to individual modules. The results in Table 7(c) are worse than ours, suggesting that displacing different parts should use the same module to transfer information between different clusters. Before incorporating the global feature into each part, we sent it to the convolutional layers. We expect convolutional layers to extract important features about part organization so that the point aggregation module can better deform point clusters according to the implicit organization. The values in Table 7(d) and (f) show that the performance can indeed be improved with the convolutional layers. In addition, we tried another architectural design. The model first generates the point cloud of the entire object and uses multiple PA modules to displace different point groups. Comparing the results in Table 7(e) and (f), the values increase a lot. To further compare the performance between these two architectures, we visualize the translation results on the chair and table dataset in the supplementary materials.

5 CONCLUSIONS

In this paper, we propose a novel parts-to-whole architecture for unpaired shape translation of 3D point clouds, which reconstructs a whole point cloud with multiple point clusters. Our method decomposes a shape feature into multiple part features, and considers both global and local features

on losses. Moreover, a part aggregation module is presented to incorporate global information and deform each point cluster for refinement. This multi-part shape representation allows our model to preserve fine local details and make the translation output more similar to the distinctive shape feature of the source shape. Experiments manifest the effectiveness and advantages of our architecture design compared to previous methods in shape transformation.

The limitation of our work is that when the number of clusters is large, multiple clusters are prone to overlap. Especially in the translation between chair and table domains, the combination of multiple parts may not always be smooth, especially on the table top. In the future, we would like to investigate how to further constrain the relationship between multiple point clusters, and we are also interested in applying such a technique for real-time applications, such as augmented reality [Huang et al. 2022; Wu et al. 2016].

REFERENCES

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012* (2015).

Qimin Chen, Johannes Merz, Aditya Sanghi, Hooman Shayani, Ali Mahdavi-Amiri, and Hao Zhang. 2022. UNIST: Unpaired Neural Implicit Shape Translation Network. In *IEEE Conference on Computer Vision and Pattern Recognition*. 18593–18601.

Shu-Yu Chen, Feng-Lin Liu, Yu-Kun Lai, Paul L Rosin, Chunpeng Li, Hongbo Fu, and Lin Gao. 2021. DeepFaceEditing: deep face generation and editing with disentangled geometry and appearance control. *ACM Transactions on Graphics (TOG)* 40, 4, Article 90 (2021), 15 pages.

Shu-Yu Chen, Wanchao Su, Lin Gao, Shihong Xia, and Hongbo Fu. 2020. DeepFaceDrawing: Deep generation of face images from sketches. *ACM Transactions on Graphics (TOG)* 39, 4, Article 72 (2020), 16 pages.

Zhiqin Chen and Hao Zhang. 2019. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 5939–5948.

Johanna Delanoy, Mathieu Aubry, Phillip Isola, Alexei A. Efros, and Adrien Bousseau. 2018. 3D Sketching Using Multi-View Deep Volumetric Prediction. *Proc. ACM Comput. Graph. Interact. Tech.* (2018), 22 pages.

Theo Deprelle, Thibault Groueix, Matthew Fisher, Vladimir Kim, Bryan Russell, and Mathieu Aubry. 2019. Learning elementary structures for 3d shape generation and matching. *Advances in Neural Information Processing Systems* 32 (2019).

Haoqiang Fan, Hao Su, and Leonidas J Guibas. 2017. 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*. 605–613.

Vignesh Ganapathi-Subramanian, Olga Diamanti, Soeren Pirk, Chengcheng Tang, Matthias Niessner, and Leonidas Guibas. 2018. Parsing geometry using structure-aware shape templates. In *2018 International Conference on 3D Vision (3DV)*. IEEE, 672–681.

Lin Gao, Jie Yang, Yi-Ling Qiao, Yu-Kun Lai, Paul L Rosin, Weiwei Xu, and Shihong Xia. 2018. Automatic unpaired shape deformation transfer. *ACM Transactions on Graphics (TOG)* 37, 6, Article 237 (2018), 15 pages.

Lin Gao, Jie Yang, Tong Wu, Yu-Jie Yuan, Hongbo Fu, Yu-Kun Lai, and Hao Zhang. 2019. SDM-NET: Deep generative network for structured deformable mesh. *ACM Transactions on Graphics (TOG)* 38, 6, Article 243 (2019), 15 pages.

Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. 2019. Learning shape templates with structured implicit functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7154–7164.

Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. 2018. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 216–224.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. 2017. Improved training of wasserstein gans. *Advances in neural information processing systems*, 5769–5779.

Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. 2021. Pct: Point cloud transformer. *Computational Visual Media* 7, 2 (2021), 187–199.

Kaveh Hassani and Mike Haley. 2019. Unsupervised multi-task feature learning on point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 8160–8171.

Wei-Lun Huang, Chun-Yi Hung, and I-Chen Lin. 2022. Confidence-Based 6D Object Pose Estimation. *IEEE Transactions on Multimedia* 24 (2022), 3025–3035.

Junho Kim, Minjae Kim, Hyeonwoo Kang, and Kwanghee Lee. 2020. U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation. In *International Conference on Learning Representations, ICLR 2020*.

Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. 2017. Learning to discover cross-domain relations with generative adversarial networks. In *International conference on machine learning*. PMLR, 1857–1865.

Vladimir G Kim, Wilmot Li, Niloy J Mitra, Siddhartha Chaudhuri, Stephen DiVerdi, and Thomas Funkhouser. 2013. Learning part-based templates from large collections of 3D shapes. *ACM Transactions on Graphics (TOG)* 32, 4, Article 70 (2013), 12 pages.

Yin-Hsuan Lee, Yu-Kai Chang, Yu-Lun Chang, I-Chen Lin, Yu-Shuen Wang, and Wen-Chieh Lin. 2018. Enhancing the Realism of Sketch and Painted Portraits With Adaptable Patches. *Computer Graphics Forum* 37, 1 (2018), 214–225.

Jun Li, Kai Xu, Siddhartha Chaudhuri, Ersin Yumer, Hao Zhang, and Leonidas Guibas. 2017. Grass: Generative recursive autoencoders for shape structures. *ACM Transactions on Graphics (TOG)* 36, 4, Article 52 (2017), 14 pages.

Kaichun Mo, Paul Guerrero, Li Yi, Hao Su, Peter Wonka, Niloy Mitra, and Leonidas J Guibas. 2019a. Structurenet: Hierarchical graph networks for 3d shape generation. *arXiv preprint arXiv:1908.00575* (2019).

Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. 2019b. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 909–918.

Chengjie Niu, Jun Li, and Kai Xu. 2018. Im2struct: Recovering 3d shape structure from a single rgb image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4521–4529.

Soomin Park, Deok-Kyeong Jang, and Sung-Hee Lee. 2021. Diverse Motion Stylization for Multiple Style Domains via Spatial-Temporal Graph-Based Generative Model. *Proc. ACM Comput. Graph. Interact. Tech.* (2021), 17 pages.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 652–660.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*. 5105–5114.

Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. 2000. The earth mover's distance as a metric for image retrieval. *International journal of computer vision* 40, 2 (2000), 99–121.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. 2017. Dynamic routing between capsules. *Advances in neural information processing systems* 30, 3859–3869.

Chun-Yu Sun, Qian-Fang Zou, Xin Tong, and Yang Liu. 2019. Learning adaptive hierarchical cuboid abstractions of 3d shape collections. *ACM Transactions on Graphics (TOG)* 38, 6, Article 241 (2019), 13 pages.

Weiwei Sun, Andrea Tagliasacchi, Boyang Deng, Sara Sabour, Soroosh Yazdani, Geoffrey E Hinton, and Kwang Moo Yi. 2021. Canonical Capsules: Self-Supervised Capsules in Canonical Pose. *Advances in Neural Information Processing Systems* 34 (2021), 24993–25005.

Yaniv Taigman, Adam Polyak, and Lior Wolf. 2016. Unsupervised cross-domain image generation. *arXiv preprint arXiv:1611.02200* (2016).

Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. 2017. Learning shape abstractions by assembling volumetric primitives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2635–2643.

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*, 6000–6010.

Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. 2019a. 3dn: 3d deformation network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1038–1046.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. 2019b. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (TOG)* 38, 5, Article 146 (2019), 12 pages.

Zongji Wang, Yunfei Liu, and Feng Lu. 2021. Cloud Sphere: A 3D Shape Representation via Progressive Deformation. *arXiv preprint arXiv:2112.11133* (2021).

Li-Chen Wu, I-Chen Lin, and Ming-Han Tsai. 2016. Augmented reality instruction for object assembly based on markerless tracking. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. ACM, 95–102.

Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. 2020. Pq-net: A generative part seq2seq network for 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 829–838.

Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. 2016. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)* 35, 6, Article 210 (2016), 12 pages.

Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. 2017. Dualgan: Unsupervised dual learning for image-to-image translation. In *Proceedings of the IEEE international conference on computer vision*. 2849–2857.

Kangxue Yin, Zhiqin Chen, Hui Huang, Daniel Cohen-Or, and Hao Zhang. 2019. LOGAN: Unpaired shape transform in latent overcomplete space. *ACM Transactions on Graphics (TOG)* 38, 6, Article 198 (2019), 13 pages.

Kangxue Yin, Hui Huang, Daniel Cohen-Or, and Hao Zhang. 2018. P2p-net: Bidirectional point displacement net for shape transform. *ACM Transactions on Graphics (TOG)* 37, 4, Article 152 (2018), 13 pages.

Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. 2021. Pointr: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*. 12498–12507.

Kaiyi Zhang, Ximing Yang, Yuan Wu, and Cheng Jin. 2022. Attention-based transformation from latent features to point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 3291–3299.

Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 2019. 3D point capsule networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1009–1018.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*. 2223–2232.

Chuhang Zou, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem. 2017. 3d-prnn: Generating shape primitives with recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 900–909.

Received 21 December 2022; revised 15 March 2023; accepted 23 March 2023