Shape Cloud Collage on Irregular Canvas

Sheng-Yi Yao, Dong-Yi Wu, Thi-Ngoc-Hanh Le, Tong-Yee Lee, *Senior Member, IEEE*

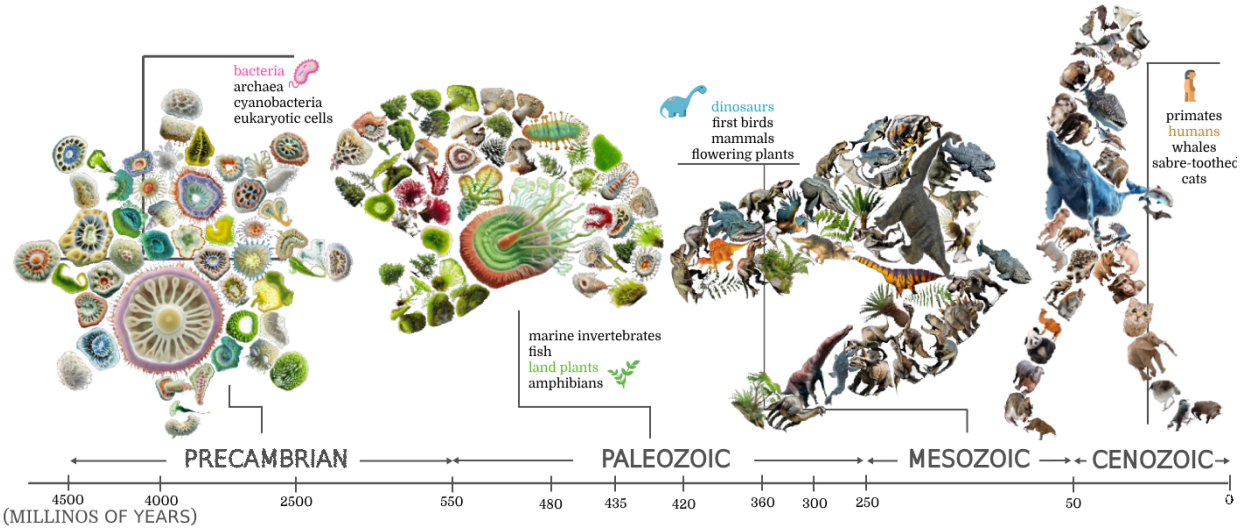


Fig. 1: Geologic timeline. We present four examples of shape cloud collages generated using our method. These visualizations depict flora and fauna from four distinct periods in Earth's history, organized and scaled according to their relative sizes.

Abstract— This paper addresses a challenging and novel problem in 2D shape cloud visualization: arranging irregular 2D shapes on an irregular canvas to minimize gaps and overlaps while emphasizing critical shapes by displaying them in larger sizes. The concept of a shape cloud is inspired by word clouds, which are widely used in visualization research to aesthetically summarize textual datasets by highlighting significant words with larger font sizes. We extend this concept to images, introducing shape clouds as a powerful and expressive visualization tool, guided by the principle that "a picture is worth a thousand words. Despite the potential of this approach, solutions in this domain remain largely unexplored." To bridge this gap, we develop a 2D shape cloud collage framework that compactly arranges 2D shapes, emphasizing important objects with larger sizes, analogous to the principles of word clouds. This task presents unique challenges, as existing 2D shape layout methods are not designed for scalable irregular packing. Applying these methods often results in suboptimal layouts, such as excessive empty spaces or inaccurate representations of the underlying data. To overcome these limitations, we propose a novel layout framework that leverages recent advances in differentiable optimization. Specifically, we formulate the irregular packing problem as an optimization task, modeling the object arrangement process as a differentiable pipeline. This approach enables fast and accurate end-to-end optimization, producing high-quality layouts. Experimental results show that our system efficiently creates visually appealing and high-quality *shape clouds* on arbitrary canvas shapes, outperforming existing methods.

Index Terms—Shape cloud, shape collage, shape packing, irregular shape packing.

1 Introduction

Data visualization has demonstrated its importance in the era of digital life particularly in the realm of visual art. Given a collection of images, we may need thousands of words to describe the insight message from the collection. Yet, visualizing information through images offers several advantages over using text alone. Images allow for rapid information processing as the ability to convey complex information more quickly than a paragraph of texts. Images create a more lasting impression, leading to improved retention and recall. Plus, visual images can transcend language barriers. By packing the collection within an image-based container (referred to as "shape cloud"), we can increase engagement, convey the importance of the information, and also construct a more immersive and impactful narrative. Visualizing information via images has been an essential tool in several applications. For example, in branding and marketing, shape cloud can be leveraged to create eye-catching visuals for products or events. In merchandise design, shape cloud can be used to allow fans to incorporate the artist's style into their daily lives through creating merchandise featuring clothing, accessories, and other products. In the realm of social media and content creation, brands and individuals could rely on the power of visual content in engaging audiences of shape cloud to create shareable and viral content. We exhibit samples of shape cloud created by artist lalan [17] in Fig. 2. While hand-designed artwork has a unique and authentic quality, this may face challenges in scalability, managing and organizing complex designs more efficiently, reproduction, or efficiency. Based on this observation, such a computer-aid function is demanded to alleviate the limitations of a hand-designed one. To this end, we investigate an optimization-based scheme that aims to create shape cloud more effectively. Our method serves as an informative way to visualize a large collection of shapes according to their relative information. For example, if we want to summarize the total number of sales of the products we are running in our business, we can collage

- *Sheng-Yi Yao is with National Cheng-Kung University. E-mail: nd8081018@gs.ncku.edu.tw.*
- *Dong-Yi Wu is with National Cheng-Kung University. E-mail: cutechubbit@gmail.com.*
- *Thi-Ngoc-Hanh Le is with International University, VNU-HCM. E-mail: ltnhanh@hcmiu.edu.vn.*
- *Tong-Yee Lee is with National Cheng-Kung University. E-mail: tonylee@mail.ncku.edu.tw.*

the image of the products guided by their number of sales. Readers can see our visualization in Fig. 7, 8, 9, 10.

(a) (b)

Fig. 2: Shape cloud by artists lalan [17] (a) and our methods (b).

Moreover, since there are advances in shape-bounded word cloud [4, 41], our shape cloud method supports visualization in bounded shape as well, e.g. a collection of cars in a car-shaped layout. We use the terms "shape" and "object" interchangeably to refer to an image patch containing an object. Users can quickly identify interesting patterns among many objects. For instance, a scientific magazine article could feature an infographic that collages species from different geological periods, arranged along a timeline based on their sizes, as illustrated in Fig. 1. This approach can captivate readers by sparking curiosity about the evolutionary processes the Earth has undergone. Additionally, it provides a clear and visually engaging overview of the dominant species in each era, which may be difficult to achieve with text-based visualizations like word clouds.

When designing shape cloud, we refer to decades of research on word cloud [2, 5, 8, 10, 30, 35]. The design space of the word cloud can be decomposed as two main visual parameters: magnitude encoding and layout [30]. The magnitude encoding is the design decision about the way we visually encode the quantitative value associated with each word. The layout problem is concerned with how to position each text on the canvas. For the magnitude encoding, it has been found that size (font size) is the most popular and effective means of encoding the data compared to other visual encoding channels like color, font, orientation, and location [10, 30]. About 92% of the real-world word clouds that appear in academia and journalism use the font size to encode the data [10]. Multiple researches [8, 30, 35] empirically show that humans are significantly more sensitive to words with larger font sizes than words with other attributes like word placement and word proximity. Based on these findings, we focus on using the objects' size to encode the quantitative data.

The irregular packing or nesting problem has been shown to be NP-complete [6]. To compactly arrange irregular objects without overlaps, existing layout strategies rely on various placement heuristics including the Archimedean spiral used in the most word cloud systems, shape matching [16, 21], physic-based simulation [32] and data-driven modeling [43, 44]. Our layout problem is more challenging than traditional packing settings for two key reasons: (1) Scalable objects: The objects in our problem are scalable, meaning their final sizes are not predetermined, unlike in previous works [16, 20, 23, 43, 44]. Additionally, we assume that objects are uniformly scaled, maintaining a certain ratio between their sizes. (2) Arbitrary container shapes: The container shape in our problem can be highly irregular, as opposed to the rectangular shapes typically assumed in existing research [20, 21, 23, 43, 44]. This combination of scalability and arbitrary container shapes makes it impractical to collect sufficient data for addressing the task in a data-driven manner. As a result, we face a *scalable irregular packing problem*, a challenge that has not been addressed in prior work.

In this work, we first visually encode each object based on a designated value, which we refer to as a "key." A key represents a specific property of an item within a given collection. For example, the key could be the number of international visitors to a given city as demonstrated in Fig. 4. We then address the *scalable irregular packing problem*, which we identify as a mixed discrete-continuous optimization problem. The discrete component typically involves placement decisions, such as determining the order or location of objects on the canvas. In contrast, the continuous component includes determining precise coordinates, rotations, and scaling factors of the objects. To enable end-to-end optimization, we transform the discrete variables into continuous space and formulate the packing problem as a differentiable image sampling task, which can be optimized using gradient-based methods. We evaluate the effectiveness of our approach through extensive testing on various shape configurations, incorporating diverse collections, shapes, and designated keys. Our results demonstrate that the proposed method is effective, efficient, and generalizable.

The main contributions of this paper are summarized as follows:

- We propose a novel visualization framework called *shape cloud* for 2D shapes that mimics the concept of the word cloud.
- We provide an analysis of the task of the associated task of mix discrete-continuous optimization.
- Our packing optimization framework is designed to tackle the challenging scalable irregular packing problem at an interactive rate without the need for extensive training.
- Experiments show that our method outperforms qualitatively and quantitatively existing tools and methods on our curated irregular packing dataset.

2 Related Work

2.1 Word Cloud

Word cloud is a famous visualization topic. Wordle [39] generates word layout by greedily filling the blank space guided by spiral curves and uses colors and fonts to enrich the visualization. Koh et al. [15] create *ManiWordle*, a system that supports interactive editing of word cloud. Strobelt et al. [37] propose *Rolled-out Wordles* that has better performance in removing overlaps. The main drawback of these two methods [15, 37] is that the overall shape is fixed. Maharik et al. [22] propose an algorithm to generate micrography, which arranges texts into beautiful shapes. Although micrography can accurately represent the shapes, it can not be used effectively as an information visualization tool due to its minuscule texts. Chi et al. [4] come up with an idea of shape-varying word cloud based on rigid body dynamics. *Edwordle* [40] is proposed as an application for consistency-preserving editing. It allows users to move and edit the texts dynamically while maintaining neighborhood relations. Wang et al. [41] propose *ShapeWordle*, an extension to Wordles, to deal with more complex canvas shapes. However, these word cloud methods usually treat words as boxes when running simulations or removing overlaps. This does not work well with the irregular shapes. In contrast, in this paper, our approach will handle irregular shapes well.

2.2 Image Collage

Automatic generation of image collages is a well-studied problem. The authors in [19, 26, 42, 45] have investigated various schemes to maintain the primary aspect ratio of images while arranging them onto a rectangular canvas. Keeping the shape of the canvas is also an effective concept, as proposed in [18, 31, 38]. In these works, the images are cropped such that the important content is preserved. Along with these research works, several commercial applications, such as Shape Collage [34], FigrCollage [36], ShapeX [29], and Adobe [1], are designed to be user-friendly, allowing users without expertise to obtain plausible results. These approaches differ from this paper mainly in two aspects. The first difference is that image collage research focuses on photos, which are rectangular. Hence, they usually do not consider the irregular shapes of the objects and the coupling of the objects' contours. Second, the canvas is usually rectangular, whereas we can deal with irregularly-shaped contours. Although there are tools that can handle irregular shape canvas, e.g. Shape Collage [34], they often achieve that by overlapping the objects. In some visual sense, it is not appealing with too many overlapped images. In contrast, our method tries to avoid overlapping as possibly as we can, particularly for irregular shapes on

irregular-contour canvas. Therefore, our problem presents a greater challenge for us to tackle.

2.3 Object Packing

The generation of shape cloud is closely related to the packing problem. The packing problem is concerned with packing geometrical shapes into a container. PAD [16] approaches the irregular shape packing problem with curve descriptor and local optimization. Ma et al. [21] propose continuous and combinatorial optimization methods to calculate the best placement of irregular shapes in three-dimensional space. These two methods can create pretty good results but the running time is too long to be used as a real-time or interactive visualization tool. Saputra et al. [32] propose a simulation-based deformation model treating objects as a mass-spring system. Since it is based on simulation, we have little control over the shape size and the final deformation result, i.e. the object might be heavily distorted or be placed upside-down. This greatly affects people's ability to understand the content of the object in visualization applications. Recently, data-driven methods [43, 44] have emerged as promising alternatives. GFPack [43] trains a gradient field to model correlations between shapes and uses this field to generate final layouts. In contrast, LISP [44] employs deep neural policies to handle high-level grouping tasks and low-level pose optimization. However, these methods face challenges in generalization when applied to our task, particularly with unseen shapes or varying object counts. They often require extensive fine-tuning for specific datasets and object configurations, with training datasets typically requiring tens of thousands of examples—making comprehensive handling of irregular shapes impractical. Additionally, unlike classic irregular packing, graphic applications demand more flexibility and diversity, allowing shapes to scale, deform, and fit within arbitrarily shaped canvases, rather than being restricted to traditional rectangular layouts.

A critical distinction between irregular shape packing problems and the shape cloud generation task lies in flexibility. Packing problems assume predefined and fixed piece sizes, while shape cloud generation allows size adjustments to optimize canvas coverage. Additionally, since packing algorithms typically do not prioritize aesthetics, they may fail to produce visually appealing layouts that are suitable for visualization purposes.

3 Problem Analysis

Solving our scalable irregular packing problem in a purely combinatorial sense requires exponential time. Especially considering the scalable aspect of our task can easily lead to combinatorial explosion. To address this challenge, we propose to model the problem in a continuous space. Specifically, the placement of an irregular shape $\boldsymbol{P} \subset \mathbb{R}^2$ onto the canvas $\boldsymbol{P}_0$ means deciding the translation vector $\boldsymbol{v}$, the rotation angle θ, and the scaling factor s. The placed shape $\boldsymbol{P}'$ is given by

$$\boldsymbol{P}' = \{s(x\cos(\theta) - y\sin(\theta), x\sin(\theta) + y\cos(\theta)) + \boldsymbol{v} \mid (x,y) \in \boldsymbol{P}\}. \quad (1)$$

The parameters governing the layout include translation vectors $\boldsymbol{v}_i$ and rotation angles θ_i for each of the N objects, along with a global scaling factor s, which is shared across all objects to maintain their relative proportions.

The goal of our optimization is that we want the objects to cover the canvas as fully as possible. Therefore, the level of coverage is defined as

$$C = \bigcup_{i=1}^{N} \boldsymbol{P}'_i \cap \boldsymbol{P}_0. \quad (2)$$

Directly optimizing the parameters (s, θ, $\boldsymbol{v}$) often leads to suboptimal solutions due to convergence to local optima. This limitation arises because these parameters do not explicitly capture the neighborhood structure, hindering their ability to effectively model the grouping and matching of objects. To address this issue, it becomes necessary to explicitly incorporate the grouping and matching of objects into the model. Furthermore, our task introduces an additional layer of complexity due to the irregularity of the canvas. Unlike traditional rectangular settings, where the canvas is relatively homogeneous and the specific location is less critical, in our case, each location on the canvas possesses unique characteristics. As illustrated in Fig. 3, in a rectangular setting, swapping objects on the left and right sides of the canvas results in a similar overall layout. However, in an irregular canvas, exchanging the positions of objects leads to a significantly different layout due to the asymmetrical canvas shape.

Thus, our problem can be formulated as a mixed discrete-continuous optimization task. The discrete component often pertains to placement decisions, such as object order or location on the canvas, while the continuous component involves determining the objects' precise coordinates, rotations, and scaling factors. Mixed discrete-continuous optimization problems are generally more challenging to solve than purely discrete or purely continuous problems, as they combine the difficulties inherent to both variable types. Consequently, they often require specialized algorithms tailored to the problem at hand. Recent advancements in deep learning, particularly in learnable discrete representations [12, 14, 24, 45], offer a promising direction by relaxing discrete variables into continuous forms, enabling end-to-end optimization. Inspired by these approaches, we aim to reformulate the discrete variables into continuous representations, allowing the entire model to be optimized jointly using gradient-based optimization techniques.

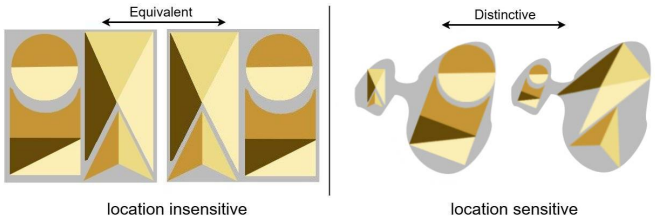


Fig. 3: Illustration of location sensitivity. Left: Swapping objects on the left and right sides of the canvas results in a similar overall layout. Right: Exchanging the positions of objects leads to a significantly different layout due to the asymmetrical canvas shape.

Finally, the hard collision constraint has to be converted into a form that is differentiable. Traditionally, the packing problem is divided into two parts: (1) objective function maximization/minimization and (2) collision detection [20]. Detecting overlaps is often carried out in a separate step different from the optimization logic. That is, the optimization algorithm first selects a possible candidate location based on some criteria and the collision detection is used to verify the feasibility of the placement. Or the other way around, the collision detection first produces a set of feasible solutions and the optimization logic selects the best option, as in the case of no-fit polygon (NFP) [3, 33]. However, there are two drawbacks to this approach. First, this trial-and-error loop is inefficient. For example, collision detection can only tell the optimization module if the placement is valid or not but provide no clue for the optimization module to adjust the placement. Second, this process is inherently sequential, meaning some pieces have to be placed first before collision detection can run and the optimization module can select the best result. This amounts to a greedy strategy which can easily result in suboptimal results.

We achieve this by softening the collision as the area overlaps and adding this into the loss function. Minimizing the collision loss amounts to resolving the collision. The level of overlaps is written as the union of all pairwise intersections and the intersection with the canvas boundary

$$O = \bigcup_{i,j=1,i<j}^{N} (\boldsymbol{P}'_i \cap \boldsymbol{P}'_j) \cup \bigcup_{i=1}^{N} (\boldsymbol{P}_0^c \cap \boldsymbol{P}'_i), \quad (3)$$

where $\boldsymbol{P}_0^c$ is the complement of the canvas shape.

Therefore, the overall optimization objective is maximizing the coverage C while minimizing overlaps O, i.e., $\arg\max_{\Theta}(C - O)$, where Θ is all the parameters of the model.

4 Our Method

4.1 Overview

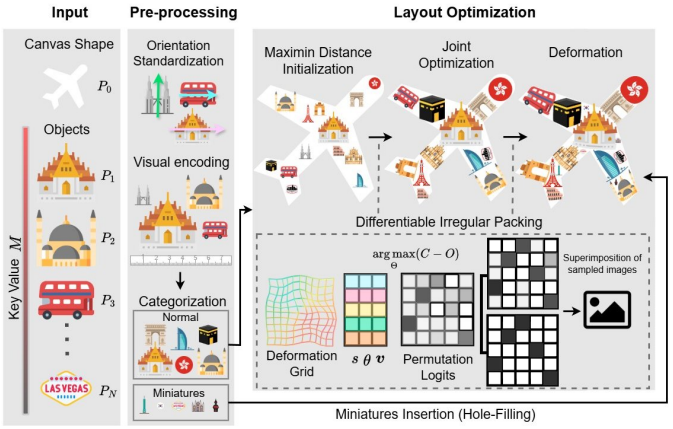


Fig. 4: Overview of our shape cloud system: The pipeline takes 2D objects (left), processes them through a pre-processing stage (middle), and optimizes the layout using differentiable irregular packing (right).

Our proposed framework is depicted in Fig. 4. The system takes as input a collection of 2D objects, their corresponding quantitative values, and an irregularly shaped canvas. The objects are oriented uniformly based on shape moments, and their sizes are visually encoded in proportion to the given quantitative values. Depending on their sizes, the objects are categorized into two groups: normal-sized and miniature. These miniature elements have negligible impact on the layout but can hinder optimization convergence due to the problem's increased size. Therefore, instead of integrating them during the main optimization phase, these miniatures are added afterward using a basic scanline hole-filling algorithm. The core of our system is a differentiable irregular packing process designed for end-to-end optimization. Initially, the layout parameters are spread maximally across the canvas. We then jointly optimize all parameters, including scaling, rotation, translation, and permutation logits. After the optimization process, miniatures are inserted into the layout, and deformation grids are added to the parameter sets to finalize the optimization process.

4.2 Object Pre-processing

Visual Encoding. Given a collection of N objects $\boldsymbol{P}_i$ where $i = 1...N$ and key values designated by the user, our goal is to encode the key values visually onto the canvas $\boldsymbol{P}_0$. The key value, which is also mentioned as "information" in our paper, is a positive value associated with each object, denoted as $M \in \mathbb{R}^+$. For example, if we assume each object represents a product on which a company is running its business, M could be the revenue ratio, percentage of total sales, etc., of a certain product. Accordingly, the object with a higher M will occupy a larger portion of the shape cloud. In a word cloud, the significance of a word is manifested in font size measuring the vertical height of a certain character. Unlike word characters in which aspect ratio is close to that of a square, object shapes can have arbitrary aspect ratios. We therefore define the shape size as the diagonal length of the bounding box enclosing the shape. To link the diagonal length with the input key values, we first normalize the key values within range $[0,1]$ by $M' = \frac{M}{M_{max}}$. Since the size of each object is proportional to its M' value, we assign the size of each object to be $s\sqrt{M'}$. We take the square root because the area of an object, and therefore the perceived size, is quadratic in terms of the diagonal length. The global multiplier s will be initialized in the layout initialization step and optimized, which will be described later.

Reference Frame Standardization. The optimization process requires that the translation and rotation of different objects be interchangeable. To achieve this, we standardize the reference frame for each object. First, the centroid of the shape is translated to the origin. Next, the shape is rotated to align its principal axes with a common orientation, determined by the eigenvectors of its moment of inertia matrix. These steps establish a consistent and uniform reference frame, enabling seamless and interchangeable transformations throughout the optimization process. From this point forward, we assume that all $\boldsymbol{P}_i$ are standardized.

Normal-Sized v.s. Miniatures. In practical applications, certain inputs may include excessively small shapes, defined as objects with an area less than m percent of the total canvas area. These shapes are excluded from direct participation in the optimization process due to their negligible impact on the overall visual appearance and the disproportionate increase in computational complexity caused by the additional dimensions. Moreover, the minuscule size of these shapes renders them perceptually equivalent to points when compared to significantly larger objects, obviating the need for fine-grained adjustments to their orientation or precise placement. Instead, these shapes are placed sequentially on the canvas by identifying the regions of maximal empty space. This empty space is measured using a distance transform of the unoccupied areas, ensuring placements that align with aesthetic principles.

4.3 Differentiable Irregular Packing

4.3.1 Maximin Distance Initialization

Initialization is a critical step for ensuring the solution converge to a decent local optimum. Our initialization technique addresses the location sensitivity inherent to the task, as previously discussed. The idea is that by distributing the initial layout in a way that maximally covers the solution space, the optimization process can explore different location combinations simply by exchanging two objects. This approach is particularly crucial for highly concave canvases, where certain regions may otherwise remain unoccupied. We begin by randomly initializing s and setting $\theta_i = 0$. Next, we maximize the minimum distance between shapes in the space to ensure objects are as evenly distributed as possible. Formally, this is expressed as:

$$\max_{\boldsymbol{v}} \min_{i \neq j} D(\boldsymbol{P}_i, \boldsymbol{P}_j), \tag{4}$$

where D is the distance between two shapes.

To approximate a solution, we use a simple greedy algorithm. The process starts by placing the first object $\boldsymbol{P}_1$ randomly within the canvas. For each subsequent object $\boldsymbol{P}_i$, it is positioned to maximize its minimum distance from all previously placed objects.

4.3.2 Packing through Image Sampling

As discussed in Section 3, our problem is highly nonlinear and involves balancing both global and local impacts, making it particularly challenging for standard off-the-shelf optimizers. Our key contribution lies in recognizing that the packing process can be effectively modeled as a multi-image sampling task. The input shapes $\{P_0, P_1, P_2, \ldots, P_N\}$ are first rasterized into $H \times W$ binary images $\{I_0, I_1, I_2, \ldots, I_N\}$. These images are then stacked to form an N-channel binary tensor $\boldsymbol{I} \in \{0,1\}^{H \times W \times N}$. The core packing process builds upon the differentiable image sampling layer introduced in the spatial transformer network by Jaderberg et al. [11]. This sampling layer begins with a grid generator T_Θ, which operates on a regular grid $G = \{(x_j^t, y_j^t) \mid j \in [1 \ldots HW]\}$, where t represents the target coordinates.

$$\begin{pmatrix} x_j^s \\ y_j^s \\ 1 \end{pmatrix} = T_\Theta(G) = \begin{bmatrix} s\cos(\theta) & -\sin(\theta) & \boldsymbol{v}_x \\ \sin(\theta) & s\cos(\theta) & \boldsymbol{v}_y \\ 0 & 0 & 1 \end{bmatrix}^{-1} \begin{pmatrix} x_j^t \\ y_j^t \\ 1 \end{pmatrix}, \tag{5}$$

where (x^s, y^s) are source coordinates for sampling and $\left[\,\cdot\,\right]^{-1}$ is the inverse matrix.

Given the source coordinates, the sampled output U can be expressed as

$$U_j^c = \sum_n^H \sum_m^W \boldsymbol{I}_{nm}^c k(x_j^s - m) k(y_j^s - n), \quad \forall j \in [1 \ldots HW] \;\; \forall c \in [1 \ldots N], \tag{6}$$

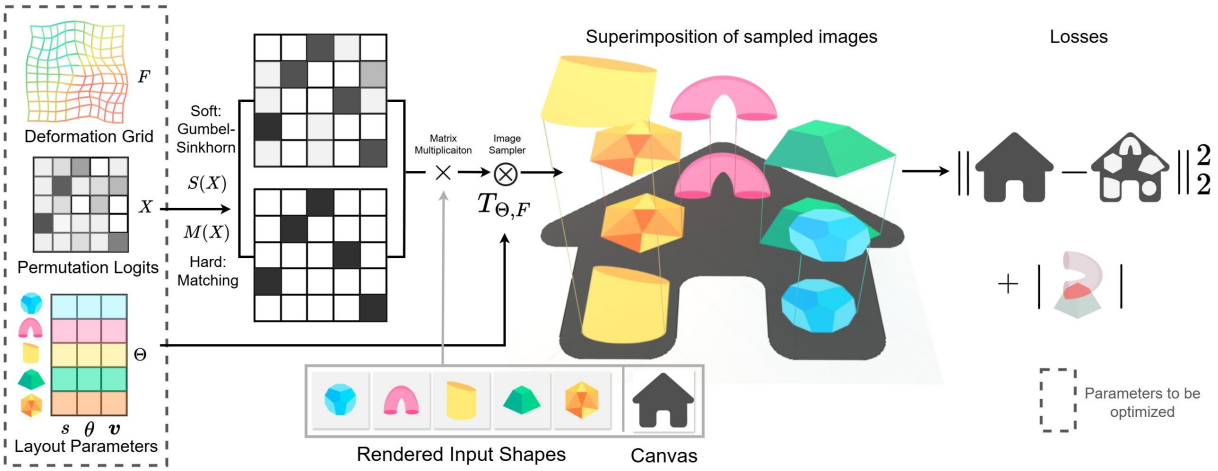


Fig. 5: Differentiable irregular packing: Optimizable parameters (dotted rectangle) are passed through a differentiable process, with the loss function (right) computed accordingly.

where $k(\cdot)$ is the sampling kernel, e.g., bilinear sampling kernel, $\boldsymbol{I}^c_{nm}$ is the value at location (n,m) in channel c and U^c_j is the output value for pixel j in channel c. This sample mechanism has been shown to be differentiable [11], i.e. the derivative of the output w.r.t. the parameter $\frac{\partial U}{\partial \Theta}$ can be derived using equation (5) and (6).

Each channel represents the transformation of an object. By superimposing results from every channel, we have the final packing layout

$$V_j = \sum_{c=1}^{N} U^c_j, \quad \forall j \in [1...HW]. \tag{7}$$

Modeling Deformation. We draw inspiration from the deformation-driven packing approach introduced in RepulsionPak [32], which demonstrates that deformation significantly enhances perceptual quality. To achieve better canvas fitting, we model objects as non-rigid shapes. Deformation is modeled as a free-form deformation grid $F = \{(x^f_k, y^f_k) \mid k \in [1...H'W']\}$, where H' and W' are grid resolution. Typically, the resolution of the deformation grid is lower than that of the original image. To align resolutions, we interpolate F to match the $H \times W$ resolution of the original image. The source coordinates are then updated to incorporate deformation as follows:

$$\begin{pmatrix} x'^s_j \\ y'^s_j \end{pmatrix} = \begin{pmatrix} x^s_j \\ y^s_j \end{pmatrix} + \alpha\sigma\left(\begin{pmatrix} x^f_j \\ y^f_j \end{pmatrix}\right), \tag{8}$$

where α is a hyperparameter that controls the extent of deformation, ranging from 0 (no deformation) to 1 (maximum deformation). The sigmoid function $\sigma(\cdot)$ ensures smooth and bounded adjustments to the deformation grid. To provide more control and preserve the recognizability of semantically meaningful shapes, we allow users to customize whether to deform each object and independently adjust the deformation strength α for each object. Users can choose to enable deformation only for less important objects (smaller M), ensuring that important shapes can maintain their original appearance while still improving canvas coverage efficiency through deformable objects.

4.3.3 Latent Permutation Representation

We explicitly model the combination of shape grouping and location as the permutation of object images I_i. This permutation is represented by a permutation matrix Π, which is a binary square matrix where each row and column sums to 1. Let the permutation to be optimized be parameterized by a square matrix $X \in \mathbb{R}^{N\times N}$, referred to as the permutation logits. To convert X into a valid permutation matrix, we apply a matching operator $M(X) = \arg\max_{\Pi\in\Pi_N} \langle \Pi, X\rangle$, where Π_N is the set of all permutation matrices of size N. However, the operator $M(X)$ is non-differentiable with respect to X due to the presence of the $\arg\max$ function.

To optimize the permutation logits X, we employ the Gumbel-Sinkhorn operator $S(X)$ [24] as our latent permutation representation. Intuitively, the Sinkhorn operator transforms a square matrix into a doubly-stochastic matrix (one in which the rows and columns sum to 1, but the elements are not necessarily binary), similar to how the softmax function operates on probability distributions. The Gumbel trick ensures that the matrix elements approximate either 0 or 1, resembling one-hot vectors.

The "soft" permutation matrix $S(X)$ is then used to permute the object collection as follows:

$$\boldsymbol{I}' = \Big[\quad S(X) \quad\Big]_{N\times N} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix}_{N\times(H\times W)}.$$

The resulting $\boldsymbol{I}'$ replaces the original images to generate the packing result. Since this process is differentiable with respect to X, the permutation logits X can be optimized jointly with Θ and F. When producing the final packing layout, we use $M(X)$ in place of $S(X)$ for a hard permutation. The fully differentiable irregular packing framework is summarized in Fig. 5. The input shapes are first rasterized into binary images and stacked into a multi-channel tensor. Shape groupings are modeled using a learnable soft permutation matrix $S(X)$, which is applied to the image tensor to reorder the shapes. Each shape is then transformed via a differentiable spatial transformation, guided by a sampling grid derived from the object's rotation angle θ, translation vector $\boldsymbol{v}$, global scaling factor s, and a learnable deformation grid F. To produce the final packing layout, the soft permutation $S(X)$ is replaced with a hard permutation matrix $M(X)$. The transformed shapes are aggregated into the final layout through image sampling, enabling end-to-end differentiable optimization. This framework allows the layout parameters to be updated via gradient-based learning.

4.4 Gradient-Based Optimization

By unifying the discrete and continuous aspects of the packing process, the scalable irregular packing problem can be reformulated as finding the optimal parameter Θ^*, F^*, and X^* to minimize the packing quality loss L. We first rewrite the coverage loss in terms of packing result $\boldsymbol{V}$ as

$$L_{coverage}(\Theta, F, X) = \text{MSE}(I_0, \boldsymbol{V}), \tag{9}$$

where MSE stands for mean square error.

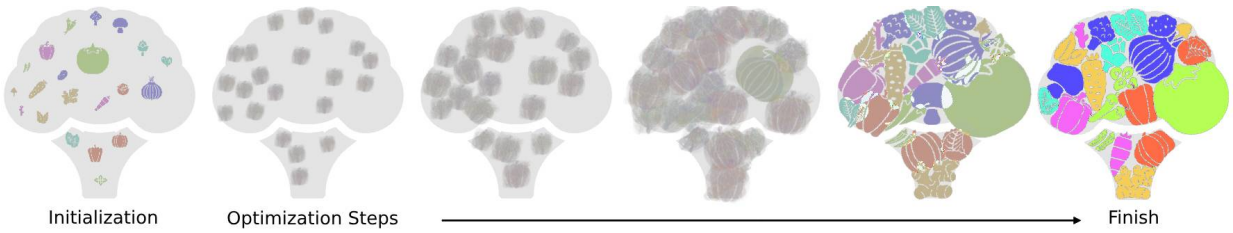


Fig. 6: This figure illustrates the joint optimization of all the layout parameters. The mixture of shapes in the early stage of the process reflects the permutation uncertainty.

Next, we minimize the overlaps based on the packing layout V_j with the following loss:

$$L_{collision}(\Theta, F, X) = \sum (\boldsymbol{V} + I_0^c)^{>1}, \tag{10}$$

where the sum counts overlapping regions and $\mathbf{X}^{>1} = \{x \mid x > 1 \text{ and } x \in \mathbf{X}\}$.

To encourage smoothness and prevent excessive deformation in the grid, we include a grid edge length regularization loss:

$$L_{deformation}(\Theta, F, X) = \frac{1}{H'W'} \sum_{i=1}^{H'} \sum_{j=1}^{W'} \left(x_{i,j}^f - x_{i+1,j}^f\right)^2 + \left(y_{i,j}^f - y_{i+1,j}^f\right)^2 + \left(x_{i,j}^f - x_{i,j+1}^f\right)^2 + \left(y_{i,j}^f - y_{i,j+1}^f\right)^2. \tag{11}$$

The packing quality loss is then: $L(\Theta, F, X) = L_{cov.} + \lambda_1 L_{col.} + \lambda_2 L_{def.}$, where λ_1 and λ_2 is the weighting for the collision term and deformation term, respectively.

In iteration t of the optimization, parameters are updated as

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta \nabla_\Theta L, F^{(t+1)} = F^{(t)} - \eta \nabla_F L, X^{(t+1)} = X^{(t)} - \eta \nabla_X L.$$

We employ a learning rate decay strategy, starting with a relatively high learning rate $\eta = 0.01$ to facilitate broad exploration of the solution space, and gradually decreasing it to $\eta = 0.001$ for more precise refinements. The optimization process is illustrated in Fig. 6.

5 Results and Discussion

5.1 Implementation Details

We implement our algorithm in PyTorch [27] to leverage its automatic differentiation engine. All the experiments are based on a PC with Intel Core i9-9900K CPU, NVIDIA GeForce RTX 2080 and 16GB RAM. We use the Adam optimizer [13] to optimize the layout parameters for 1500 iterations. Our packing algorithm implementation is highly efficient and only requires a few seconds to optimize the layout. Compared to existing tools and methods [16, 28, 32], our method is up to ten times faster. The parameters are optimized under the resolution $H \times W = 300 \times 300$ and the final results can be rendered under arbitrary resolution. The results shown in this paper are 512×512 pixels. We choose the resolution based on experience as a balance between layout quality and computational efficiency. A resolution setting of 300×300 provides sufficient spatial precision to capture object shapes while maintaining fast optimization speeds. Although increasing the resolution may improve slightly for objects with fine structures, it will significantly increase memory usage and convergence time; Conversely, setting the resolution too low can lead to loss of structural details and inaccurate overlap estimation. Therefore, we chose this resolution as a reasonable compromise in our applications. Additional analysis and visual comparisons at different resolutions are provided in the supplemental materials.

5.2 Experimental Results

Shape Cloud enables us to quantitatively visualize objects and discover patterns, which is not supported by existing word cloud-like methods and tools. Fig.7 shows an example of applying Shape Cloud to data visualization tasks and generating interesting infographics regarding the exports of three countries in 2021. Each category of exports is represented by a representative object and the size of each object reflects the share of total exports. Many interesting patterns can be observed: Three countries have very different export goods, natural resources for Brazil, apparel for Sri Lanka and oil and machinery for the United States. It can also be observed that the exports for Sri Lanka are more concentrated in one category compared to Brazil and the United States. All this information is packaged inside beautiful territory-shaped canvases. In Fig. 8, we demonstrate another application for shape cloud. The visualization shows the popularity of each product in two categories on *amazon.com*: Home and Toy. Each object in the graph corresponds to an item sold online and the size represents the popularity ranking. We can quickly see what products are trending. The product is not easy to convey using only text. It is more straightforward to use images. We can quickly and intuitively identify best-selling products. It is not possible to use text visualization tools like word cloud because we often need several words to specify a particular product. Even if you can describe a product accurately with texts, it is still far more intuitive with pictures. Like word cloud, the shape cloud is also visually pleasing and ready to be used in reports or publications.

Fig. 7: Shape Cloud visualization of 2021 exports for three countries: Brazil, Sri Lanka and the United States. The size of the object reflects the given category's percentage of total exports. The canvas shape is the black image on the left.

Fig. 8: Shape Cloud visualization of top-selling products from two categories on *amazon.com*. The larger the size the higher the rank. The canvas shape is the black image on the left.

Precise object numbers and object size. Being able to precisely control the number of objects and their size is the major difference between shape cloud and other shape visualization techniques [16, 25, 32, 34]. The design goals of previous methods, which are geared toward computer graphics applications, differ from the visualization purpose

in this paper. We put our emphasis on making sure the information is correctly encoded in the visualization. In contrast, many other tools [16, 25, 32] are designed for artistic purposes. These methods usually do not consider the content of the visualization and put more emphasis on aesthetics. For example, RepulsionPak [32] packs shapes into a composition, called *element packing*. The element in the composition is usually repeated several times and the generated results usually have no meaning other than aesthetic purposes.

Flexibility in visualizing different aspects of the data. Because we design the key to be general, our system can be used to visualize a single collection with different keys, which can generate interesting patterns. For example, Fig. 9 visualizes the NBA teams from three different perspectives, i.e., winning percentage, points per game, and opponent points per game. Although Milwaukee Bucks was ranked first overall (PCT), it was far from scoring the most. On the other hand, Sacramento Kings scored the most but fell behind in the other two respects. As demonstrated by Fig. 9, many useful insights can be extracted by comparing different shape cloud visualizations next to each other.

Fig. 9: Visualization of NBA teams from three different aspects in the year 2023. PCT: winning percentage. PPG: points per game. OPP PPG: opponent points per game.

Comparison with word cloud-like methods. We note that although some word cloud methods like ShapeWordle [41] support shape-bounded canvas, they are not particularly well-suited for data visualization tasks. For example, in Fig. 10, we compare our method and the result generated with ShapeWordle. The visualization consisted of 25 most-used social media platforms. While our shape cloud accurately recreates the canvas shape, ShapeWordle struggles to fully fill the canvas with 25 words, and the words appear relatively uniform in size. This is because each word is confined to a long rectangular shape, which restricts enlargement and compact placement. ShapeWordle relies on so-called *filling words*, which are the repeated words of the original 25 words, to fill the gap. Consequently, word cloud-like methods are sub-optimal for visualizing data on irregular canvases due to their limited size adjustability and reliance on non-informative elements. Moreover, our image-based method offers a more intuitive and effective way to explore datasets as illustrated in Fig. 10.

Comparison with PAD. Kwan et al. [16] propose a scale-invariant measurement PAD for comparing shapes. Then they employ the PAD value to create collages of shapes. However, the process requires evaluating all pairs of partial shapes of every object, which is very time-consuming. Depending on the object counts and shape resolution, the running time can easily reach a few hours even for a moderately complex collage (please refer to the PAD paper for more time statistics). This makes it unsuitable for our design goal of serving as a data visualization tool for interactive data discovery. In contrast, our packing framework demonstrates linear scalability with problem size, representing a significant improvement over PAD and other combinatorial optimization methods.

Comparison with RepulsionPak. RepulsionPak [32] is a state-of-the-art 2D artistic composition tool for generating element packing. The layout process is a physics simulation with each element represented as a deformable object. The main advantage of this method is that the simulation process produces a natural compact layout leading to a pleasant viewing experience. The downside of applying RepulsionPak method for our task is that the size of the object can not be precisely controlled. In the simulation process, every object is first scaled down to a point. Then the objects grow simultaneously until the canvas is filled. In the original implementation, all the objects grow at the same rate. We modify the code of RepulsionPak by taking into account different growth rates for each object, i.e. faster growth rates for objects with larger M. Even with this modification, the final size of the object is largely determined by the initial condition of the simulation and the interaction among other objects. For example, in Fig. 11(a), the soccer object, which is supposed to be the largest, is initialized on the left segment. Due to the space restriction, it can only grow to a size smaller than some objects. Furthermore, objects tend to grow to relatively even size since objects are competing against each other for space. In contrast, our result in Fig. 11(b) demonstrates that we can not only produce compact results but also present the data value accurately.

Fig. 10: Visualization of 25 social media platforms with the most monthly active users. (a) Our shape cloud representation. (b) ShapeWordle [41] representation.

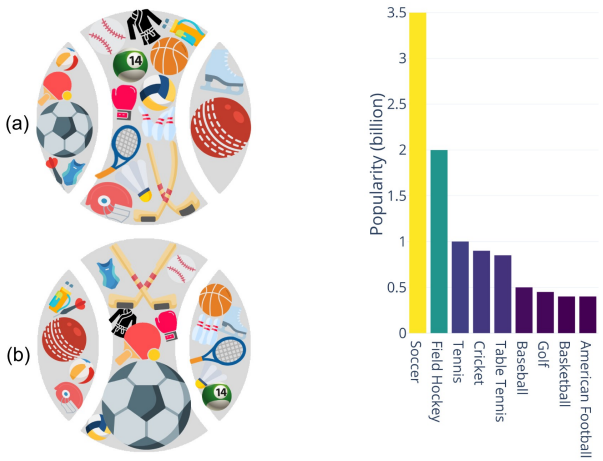


Fig. 11: Visualization of the world's most popular sports. (a) The result generated by RepulsionPak [32]. (b) The result generated by our method. The data we used to generate the visualization is shown on the right.

5.3 Evaluation

In this section, we quantitatively compare our method with three baseline methods. The three baseline methods are Shape Collage [34], RepulsionPak [32] and Deepnest [28]. These methods are commonly used methods for packing irregular shapes. Shape Collage [34] is a

(a) Shape Collage (b) RepulsionPak (c) Deepnest (d) Ours w/o deformation (e) Ours w/ deformation

Fig. 12: Qualitative comparison of four different methods, i.e. Shape Collage [34], RepulsionPak [32], Deepnest [28] and Ours.

widely used commercial tool for collage generation. Given an input image collection, Shape Collage aims to distribute the input shape evenly on the canvas by treating images as a spring-mass system. Note that the collisions between images are not explicitly detected, so many overlaps are expected. RepulsionPak [32] is modified as discussed in the previous section, i.e. independent growth speed for each object. Deepnest [28] is an open-source software for solving industrial nesting problems on laser cutters and CNC machines. In this case, the sizes of the objects are pre-defined and will not change during the optimization process. To acquire a layout comparable to ours, we run the program several times. Each time we incrementally increase the size of the objects proportionally until some objects cannot fit the canvas. The optimization mode is set to irregularly shaped canvas and the number of rotation steps is set to 12, i.e., every 30 degree. These methods are compared in terms of layout quality and data representation error. A good object layout should be able to recreate the input canvas shape as completely as possible. We quantitatively define this metric as the coverage ratio, i.e. the percentage of the canvas covered by the objects defined as

$$\text{Layout Quality} = \frac{C}{Area(P_0)}, \tag{12}$$

where C is the coverage in Equation (2). To measure how accurately data is represented with objects' size, we propose a data representation error metric defined as

$$\text{Data Representation Error} = \text{MAE}(M', \hat{M}), \tag{13}$$

where MAE stands for mean absolute error and M' is the target size and the actual size in the final visualization $\hat{M}$.

We also measure the execution time for each method. Running time is a major concern for our application. Methods that have lower running times are more suitable for data visualization tasks. The experiments are conducted with our dataset. To our best knowledge, there is no publicly available dataset specifically targeting our task, i.e., scalable irregular packing. To address this, we have curated a dataset ourselves. The dataset currently includes fifty shape collections and the corresponding key values for this experiment. All shapes and the canvases in the dataset are irregular, featuring objects from various categories such as animals, fast food, and fruits sourced from an online database [7]. The collection sizes range from 25 to 40.

Fig. 12 and Table. 1 show the comparison of all the methods along with the three metrics. In terms of visual quality, Shape Collage [34] is the worst among all four methods largely due to the smaller average size. RepulsionPak [32] and Deepnest [28] achieve good levels of visual quality as indicated by the layout quality score and the results. However, they lack something that is critical for the shape cloud visualization task. For RepulsionPak [32], it cannot precisely control the size of each object even though we give each object a different growth rate. This is because each object's growth terminates at a different time depending on other objects' growth and the initial condition. We are unable to constrain objects' size without affecting the overall packing quality. For Deepnest [28], the workflow includes a trial-and-error loop that is time-consuming. Furthermore, nesting algorithms usually adopt heuristics that place objects in one direction, e.g., from top-right to bottom-left. Large objects can clump together at one corner as shown in Fig. 12 where all large objects are placed on the top. This reduces the size diversity and leads to bad visual effect. Our method is best suited for the shape cloud application since the layout quality is the highest, the data representation error is close to zero and the execution time is much lower than either RepulsionPak [32] or Deepnest [28]. Depending on the use case, our method without deformation can produce good layouts with each object authentically represented. Our method with deformation can have even better layout quality at the expense of some distortion.

Table 1: Performance comparison of four different methods. We report the quantitative metrics in terms of Layout Quality. Layout Quality ranges from 0% to 100%

	Shape Collage	RepulsionPak	Deepnest	Ours w/o deformation	Ours w/ deformation
Layout Quality ↑	58.2%	71.4%	70.4%	75.9%	86.6%
Data Representation Error ↓	0.2852	0.081	0	0	0.0007
Time (s) ↓	2	71.8	124.5	22.8	22.6

Comparison with state-of-the-art rectangular packing. GFPack [43] and LISP [44] are the two recent advances in learning-based irregular shape packing for rectangular containers. For a fair comparison, we limit our model to operate within rectangular containers. Notably, GFPack [43] assumes no rotation of shapes, while LISP [44] allows free rotation. Consequently, we conduct experiments in two distinct settings: (1) Setting 1: In this scenario, we disable rotation parameters, set the canvas to rectangles, and compare the coverage ratio with GFPack [43]. We follow the dataset preparation described in their paper and evaluate our method using three rectangular containers with different aspect ratios. (2) Setting 2: Here, shapes can be freely rotated, and we compare our method with LISP [44] using its three benchmark datasets. The results, presented in Table 2, demonstrate that although our method is not specifically designed for rectangular containers, it achieves comparable or superior packing quality. This performance is notable as our method requires neither extensive training data nor significant training time and does not suffer from generalization issues. Additionally, the experiments highlight our method's adaptability to varying layout constraints without retraining. For instance, the rotation angle can be easily constrained within a specified range, such as $\pm 45^{\circ}$ demonstrating our method's flexibility and practicality. In our supplementary file, we show our result comparison with those of [43, 44].

Table 2: Comparison with SOTA.

	Dataset	Ours	LISP [44]	GFPack [43]
Setting 1	Dental	70.37%	-	63.20%
	Garment	76.95%	-	68.16%
Setting 2	Building	83.18%	86.28%	-
	Object	76.05%	69.74%	-
	General	71.85%	75.99%	-

5.4 Ablation Analysis

To assess the contributions of individual components in our model, we conducted ablation experiments on our prepared dataset. The results are summarized in Table 3.

Joint Optimization. The effectiveness of joint optimization, the central characteristic of our approach, is evaluated through progressively complex settings: (A) Independent Updates: In this simplest setting, we update only θ, s, and $\boldsymbol{v}$ independently, meaning each parameter is updated sequentially without considering interactions. This approach often results in suboptimal solutions due to the lack of coordination between parameters. (B) Adding Latent Permutation: Introducing the latent permutation X improves the solution quality by incorporating more variables. However, the absence of joint consideration of interactions among layout variables continues to limit the overall effectiveness. (C) Partial Joint Optimization: We divide the parameters into two independent groups: jointly optimizing θ, s, and $\boldsymbol{v}$ while handling X separately. This setup significantly improves solution quality but still falls short of our full model (Table. 3 F). We observe that simultaneously optimizing all parameters—θ, s, $\boldsymbol{v}$, and X—is critical for achieving the best layout quality. This comprehensive approach enables the model to fully exploit interactions among variables, leading to superior results.

Component Analysis. We further evaluate the impact of two critical components: Maximin initialization and the latent permutation mechanism. Both the maximin distance initialization and latent permutation mechanisms are crucial for guiding the model toward better solutions. Without maximin distance initialization, the solution is more likely to get trapped in local minima, leaving parts of the canvas insufficiently explored. Similarly, removing latent permutation increases the model's reliance on the quality of the initialization. Poor initialization can lead to suboptimal solutions, resulting in lower layout quality on average. Both components play a particularly vital role for highly irregular canvases. Maximin initialization ensures comprehensive exploration, while latent permutation allows the model to escape local minima and better navigate the solution space. Together, they are instrumental in enabling the model to achieve globally optimal layouts, especially under challenging conditions.

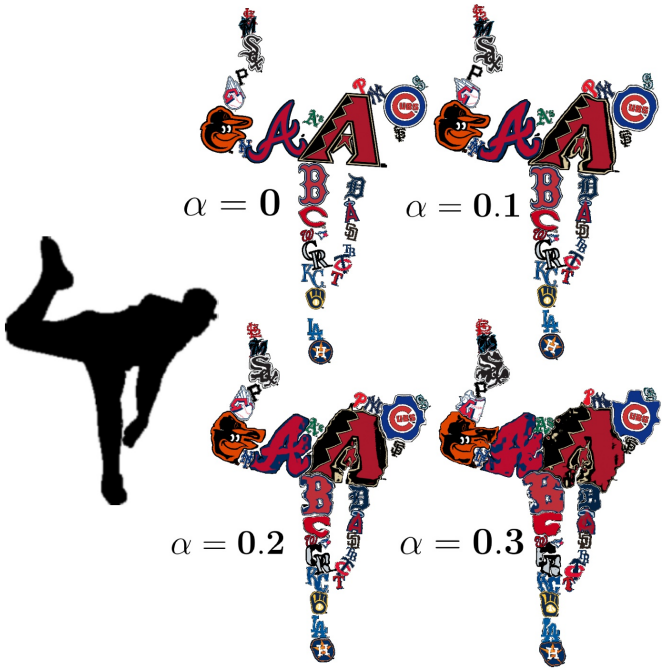


Fig. 13: Results generated with various deformation level α. The canvas shape is a pitcher pitching and is shown on the left.

Deformation v.s. Without Deformation. Allowing the deformation of the shapes is a key factor affecting the layout quality (Table. 3 F and G). Fig. 13 illustrates the effect of deformation. Without it ($\alpha = 0$), the canvas shape cannot be easily seen and the empty space within canvas has a negative impact on the aesthetics. By allowing the shapes to be deformed, the canvas shape can be truly represented. However, deformation can impair the readability of the content of each shape. For example, some of the team logos cannot be identified due to distortion

in the case of $\alpha = 0.3$. In practice, deformation settings should be customized to fit certain scenarios. When the readability of individual shapes is the priority, the α should be set to a lower value. If the canvas shape is the main focus, α should be set to a higher value. In this work, we support deformation and different deformation levels to suit various use cases.

Table 3: Ablation Study

	Methods	Layout Quality
Joint Opt.	A. $\theta \mid s \mid \boldsymbol{v}$	43.72%
	B. $\theta + X \mid s + X \mid \boldsymbol{v} + X$	54.67%
	C. $\theta + s + \boldsymbol{v} \mid X$	61.15%
Comp.	D. w/o maximin init.	61.36%
	E. w/o latent permutation	63.98%
	F. Full Model	65.79%
	G. Full Model w/ deformation $\alpha = 0.2$	81.85%

5.5 User Study

This user study is designed to evaluate user perceptions of aesthetics and data accuracy in a data visualization, with the goal of identifying the strengths and weaknesses of our methods. A diverse group of 40 participants with varying levels of experience in interpreting data visualizations was recruited for the study. Participants were given ample time to explore the data visualization thoroughly before proceeding to the tasks.

The study comprises three experiments aimed at capturing both quantitative and qualitative feedback. In the first experiment, participants were asked to rate our shape cloud using a standard aesthetic evaluation scale for visual data representations [9]. The results are summarized in Table 4 (1: strongly disagree, 5: strongly agree). Overall, our visualization design received high scores for aesthetics. Notably, it scored the highest in the Appealing category, indicating that our result successfully attracted the interest and attention of the participants.

Table 4: Aesthetic Evaluation

Category	Mean Rating	Standard Deviation
Enjoyable	4.2	0.8
Likeable	3.9	1.1
Pleasing	4.0	0.7
Nice	3.8	0.9
Appealing	4.4	0.9

In the second user study, we compare our method with three existing approaches in terms of aesthetics. Specifically, participants performed direct visual comparisons between pairs of results from our method, Shape Collage (SHP) [34], RepulsionPak (RPK) [32], and Deepnest [28]. The comparative results are presented in Table 5.

Table 5: User Preference Evaluation.

	Wins	Losses	Draws
Ours v.s. SHP [34]	80%	15%	5%
Ours v.s. RPK [32]	72%	13%	15%
Ours v.s. Deepnest [28]	86%	12%	2%

Finally, we quantitatively evaluate the accuracy of data representation in our shape cloud. In this experiment, participants were presented with our shape cloud visualization alongside four sets of data visualized using traditional bar charts. Each bar chart represented a potential underlying dataset for the shape cloud. Participants were tasked with identifying the bar chart that most accurately matched the data encoded in our shape cloud. Participants were instructed to carefully analyze both the shape cloud and the bar charts before making their selection. The accuracy of participants' choices—defined as the percentage of correct matches between the shape cloud and the corresponding bar chart—is summarized in Table 6. This metric provides a direct measure of how effectively the shape cloud conveys quantitative information. Overall, the shape cloud received positive feedback. However, a key observation emerged: when the differences in object proportions within the canvas are pronounced, users found it easier to identify the correct dataset. Conversely, when these differences were minimal, users struggled to make accurate selections. This suggests that enhancing the visual distinction between objects in the shape cloud could further improve its effectiveness in conveying data relationships.

Table 6: Data Accuracy.

ID	Collage	%	ID	Collage	%	ID	Collage	%
(a)	animal_1	85	(d)	fastfood_1	87	(g)	fruit_1	92
(b)	animal_2	81	(e)	fastfood_2	82	(h)	fruit_2	90
(c)	animal_3	78	(f)	fastfood_3	84	(i)	fruit_3	86

5.6 Limitations

In rare cases many of the objects have very thin parts, our algorithm might fail to detect these overlaps. For example in Fig. 14, many of the objects have thin wire components. The region indicated by the yellow square contains several overlapped wires. Since the area of these wires is small, the overlaps between each of these wires are often ignored by the optimization algorithm. Thin structures are generally hard to deal with irrespective of the method. One way to address this issue is to add a little thickness to the thin structure such that the area of the wire becomes more significant.

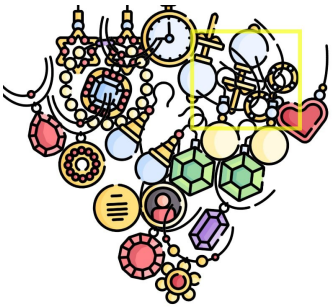


Fig. 14: The layout algorithm fails to detect overlaps in the region where the object has thin structures.

Additionally, our approach can occasionally result in layouts with slight overlaps. While these are negligible and generally imperceptible in our use case, they may pose problems in mission-critical applications such as industrial design—e.g., laser cutting, CNC, or plasma cutting—where precision is paramount. In such contexts, although our method can help optimize material layout efficiency, it should not be used as the final output, since even minor overlaps are considered artifacts in manufacturing.

6 Conclusion

In this paper, we introduce a new visualization application called *shape cloud* and propose a method to efficiently generate it. The key to the

fast generation of shape cloud is our gradient-based layout optimization algorithm. The algorithm is based on the reformulation of the packing problem as an image sampling process, which is end-to-end differentiable. We demonstrate the effectiveness of our method by comparing the state-of-the-art methods with ours. The results show that our approach performs the best in striking a good balance between visual effects and data representation accuracy compared to other related works and commercial tools. In the future, we plan to investigate how to represent the data accurately when the canvas is narrow. Another possible future work is to explore dynamic visualization of shape cloud.

Acknowledgments

The work was supported by the National Science and Technology Council, Taiwan under Grant No. 113-2221-E-006-161-MY3 and 111-2221-E-006-112-MY3.

References

[1] Adobe. Photo collage. Available: https://www.adobe.com/express/create/photo-collage, 2021. 2

[2] E. Alexander, C.-C. Chang, M. Shimabukuro, S. Franconeri, C. Collins, and M. Gleicher. Perceptual biases in font size as a data encoding. *IEEE transactions on visualization and computer graphics*, 24(8):2397–2410, 2017. 2

[3] R. C. Art Jr. *An approach to the two dimensional irregular cutting stock problem.* PhD thesis, Massachusetts Institute of Technology, 1966. 3

[4] M.-T. Chi, S.-S. Lin, S.-Y. Chen, C.-H. Lin, and T.-Y. Lee. Morphable word clouds for time-varying text data visualization. *IEEE transactions on visualization and computer graphics*, 21(12): 1415–1426, 2015. 2

[5] C. Felix, S. Franconeri, and E. Bertini. Taking word clouds apart: An empirical investigation of the design space for keyword summaries. *IEEE transactions on visualization and computer graphics*, 24(1):657–666, 2017. 2

[6] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are np-complete. *Information Processing Letters*, 12(3):133–137, 1981. ISSN 0020-0190. doi: https://doi.org/10.1016/0020-0190(81)90111-3. URL https://www.sciencedirect.com/science/article/pii/0020019081901113. 2

[7] Freepik. Flaticon. Available: https://www.flaticon.com/, 2024. 8

[8] M. J. Halvey and M. T. Keane. An assessment of tag presentation techniques. In *Proceedings of the 16th international conference on World Wide Web*, pages 1313–1314, 2007. 2

[9] T. He, P. Isenberg, R. Dachselt, and T. Isenberg. Beauvis: A validated scale for measuring the aesthetic pleasure of visual representations. *IEEE Transactions on Visualization and Computer Graphics*, 29(1):363–373, 2023. doi: 10.1109/TVCG.2022.3209390. 10

[10] R. M. Hicke, M. Goenka, and E. Alexander. Word clouds in the wild. In *2022 IEEE 7th Workshop on Visualization for the Digital Humanities (VIS4DH)*, pages 43–48. IEEE, 2022. 2

[11] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015. 4, 5

[12] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 3

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[14] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel. Neural relational inference for interacting systems, 2018. URL https://arxiv.org/abs/1802.04687. 3

[15] K. Koh, B. Lee, B. Kim, and J. Seo. Maniwordle: Providing flexible control over wordle. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1190–1197, 2010. 2

[16] K. C. Kwan, L. T. Sinn, C. Han, T.-T. Wong, and C.-W. Fu. Pyramid of arclength descriptor for generating collage of shapes. *ACM Trans. Graph.*, 35(6):229–1, 2016. 2, 3, 6, 7

[17] lalan. lalan's shutterstock page. Available: https://www.shutterstock.com/g/aalto, 2024. 1, 2

[18] L. Liu, H. Zhang, G. Jing, Y. Guo, Z. Chen, and W. Wang. Correlation-preserving photo collage. *IEEE transactions on visualization and computer graphics*, 24(6):1956–1968, 2017. 2

[19] S. Liu, X. Wang, P. Li, and J. Noh. Trcollage: efficient image collage using tree-based layer reordering. In *2017 International Conference on Virtual Reality and Visualization (ICVRV)*, pages 454–455. IEEE, 2017. 2

[20] E. López-Camacho, G. Ochoa, H. Terashima-Marín, and E. K. Burke. An effective heuristic for the two-dimensional irregular bin packing problem. *Annals of Operations Research*, 206:241–264, 2013. 2, 3

[21] Y. Ma, Z. Chen, W. Hu, and W. Wang. Packing irregular objects in 3d space via hybrid optimization. *Computer Graphics Forum*, 37(5):49–59, 2018. doi: https://doi.org/10.1111/cgf.13490. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.13490. 2, 3

[22] R. Maharik, M. Bessmeltsev, A. Sheffer, A. Shamir, and N. Carr. Digital micrography. *ACM Trans. Graph.*, 30(4), jul 2011. ISSN 0730-0301. doi: 10.1145/2010324.1964995. URL https://doi.org/10.1145/2010324.1964995. 2

[23] A. Martinez-Sykora, R. Alvarez-Valdés, J. A. Bennell, R. Ruiz, and J. M. Tamarit. Matheuristics for the irregular bin packing problem with free rotations. *European Journal of Operational Research*, 258(2):440–455, 2017. 2

[24] G. Mena, D. Belanger, S. Linderman, and J. Snoek. Learning latent permutations with gumbel-sinkhorn networks. *arXiv preprint arXiv:1802.08665*, 2018. 3, 5

[25] Y. Nagata and S. Imahori. Escherization with large deformations based on as-rigid-as-possible shape modeling. *ACM Transactions on Graphics (TOG)*, 41(2):1–16, 2021. 6, 7

[26] X. Pan, F. Tang, W. Dong, C. Ma, Y. Meng, F. Huang, T.-Y. Lee, and C. Xu. Content-based visual summarization for image collections. *IEEE Transactions on Visualization and Computer Graphics*, 27(4):2298–2312, 2019. 2

[27] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 6

[28] J. Qiao. Deepnest. Available: https://deepnest.io/. 6, 7, 8, 10

[29] Reasyze. Shapex. Available: https://www.reasyze.com/shapex/. 2

[30] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen. Getting our head in the clouds: toward evaluation studies of tagclouds. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 995–998, 2007. 2

[31] C. Rother, L. Bordeaux, Y. Hamadi, and A. Blake. Autocollage. *ACM transactions on graphics (TOG)*, 25(3):847–852, 2006. 2

[32] R. A. Saputra, C. S. Kaplan, and P. Asente. Improved deformation-driven element packing with repulsionpak. *IEEE Transactions on Visualization and Computer Graphics*, 27(4):2396–2408, 2021. doi: 10.1109/TVCG.2019.2950235. 2, 3, 5, 6, 7, 8, 10

[33] G. Scheithauer and J. Terno. Modeling of packing problems. *Optimization*, 28(1):63–84, 1993. 3

[34] B. Schoonhoven. Shape collage. Available: https://www.collage.nl/shapeCollage, 2023. 2, 6, 7, 8, 10

[35] J. Schrammel, S. Deutsch, and M. Tscheligi. Visual search strategies of tag clouds-results from an eyetracking study. In *IFIP Conference on Human-Computer Interaction*, pages 819–831. Springer, 2009. 2

[36] SilkenMermaid. Figrcollage. Available: https://www.figrcollage.com/. 2

[37] H. Strobelt, M. Spicker, A. Stoffel, D. Keim, and O. Deussen. Rolled-out wordles: A heuristic method for overlap removal of 2d data representatives. In *Computer Graphics Forum*, volume 31, pages 1135–1144. Wiley Online Library, 2012. 2

[38] L. Tan, Y. Song, S. Liu, and L. Xie. Imagehive: Interactive content-aware image summarization. *IEEE Computer Graphics and Applications*, 32(1):46–55, 2011. 2

[39] F. B. Viegas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. *IEEE transactions on visualization and computer graphics*, 15(6):1137–1144, 2009. 2

[40] Y. Wang, X. Chu, C. Bao, L. Zhu, O. Deussen, B. Chen, and M. Sedlmair. Edwordle: Consistency-preserving word cloud editing. *IEEE transactions on visualization and computer graphics*, 24(1):647–656, 2017. 2

[41] Y. Wang, X. Chu, K. Zhang, C. Bao, X. Li, J. Zhang, C.-W. Fu, C. Hurter, O. Deussen, and B. Lee. Shapewordle: tailoring wordles using shape-aware archimedean spirals. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):991–1000, 2019. 2, 7

[42] Z. Wu and K. Aizawa. Picwall: Photo collage on-the-fly. In *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 1–10. IEEE, 2013. 2

[43] T. Xue, M. Wu, L. Lu, H. Wang, H. Dong, and B. Chen. Learning gradient fields for scalable and generalizable irregular packing. In *SIGGRAPH Asia 2023 Conference Papers*, SA '23, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400703157. doi: 10.1145/3610548.3618235. URL https://doi.org/10.1145/3610548.3618235. 2, 3, 9

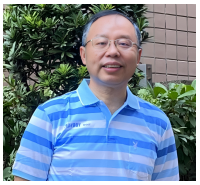
[44] Z. Yang, Z. Pan, M. Li, K. Wu, and X. Gao. Learning based 2d irregular shape packing. *ACM Trans. Graph.*, 42(6), dec 2023. ISSN 0730-0301. doi: 10.1145/3618348. URL https://doi.org/10.1145/3618348. 2, 3, 9

[45] J. Yu, L. Chen, M. Zhang, and M. Li. Softcollage: A differentiable probabilistic tree generator for image collage. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3729–3738, 2022. 2, 3

Sheng-Yi Yao received the Bachelor of Information Management degree in Department of Information Management from National Kaohsiung University of Applied Sciences, Kaohsiung, Taiwan, in 2017, and the Master of Information Management degree in Institute of Information Management from National Kaohsiung University of Science and Technology, Kaohsiung, Taiwan, in 2019, and he is working towards the Ph.D. degree at the Computer Graphics Laboratory, National Cheng Kung University, Tainan, Taiwan. His research interest is computer graphics.

Dong-Yi Wu received his BSc degree in Physics from National Cheng Kung University, Tainan, Taiwan in 2015. He is working towards the PhD degree at Computer Graphics Laboratory, Department of Computer Science and Information Engineering, National Cheng Kung University. His research interests include computer graphics, visualization and data mining.

Thi-Ngoc-Hanh Le received the PhD degree in computer science from National Cheng Kung University, Taiwan, in 2023. She has been working as a Postdoctoral fellow and a core member of the Computer Graphics Group in the Visual System Laboratory at National Cheng-Kung University, Taiwan. Currently, she is a faculty at the School of Computer Science and Engineering, International University, VNU-HCM, Vietnam. Her research interests include computer graphics, visualization and multimedia generation.



Tong-Yee Lee received the PhD degree in computer engineering from Washington State University, Pullman, in May 1995. He is currently a chair professor with the Department of Computer Science and Information Engineering, National Cheng-Kung University, Tainan, Taiwan, ROC. He leads the Computer Graphics Laboratory, National Cheng-Kung University (http://graphics.csie.ncku.edu.tw). His current research interests include computer graphics, nonphotorealistic rendering, medical visualization, virtual reality, and media resizing. He is a senior member of the IEEE Computer Society and a member of the ACM. He also serves on the editorial boards of the IEEE Transactions on Visualization and Computer Graphics.