ShapeWordle: Tailoring Wordles using Shape-aware Archimedean Spirals

Yunhai Wang, Xiaowei Chu, Kaiyi Zhang, Chen Bao, Xiaotong Li, Jian Zhang, Chi-Wing Fu, Christophe Hurter, Bongshin Lee, and Oliver Deussen

Fig. 1. Results produced by our ShapeWordle: (left) using the abstracts of VAST, InfoVis, and SciVis papers of IEEE VIS 2018 to fill an expressive shape of "VIS" and (right) words of the call for papers of IEEE VIS 2019 filled in the shape of the Canadian Maple Leaf.

Abstract—We present a new technique to enable the creation of shape-bounded Wordles, we call *ShapeWordle*, in which we fit words to form a given shape. To guide word placement within a shape, we extend the traditional Archimedean spirals to be shape-aware by formulating the spirals in a differential form using the distance field of the shape. To handle non-convex shapes, we introduce a multi-centric Wordle layout method that segments the shape into parts for our shape-aware spirals to adaptively fill the space and generate word placements. In addition, we offer a set of editing interactions to facilitate the creation of semantically-meaningful Wordles. Lastly, we present three evaluations: a comprehensive comparison of our results against the state-of-the-art technique (WordArt), case studies with 14 users, and a gallery to showcase the coverage of our technique.

Index Terms—Wordle, Archimedean spiral, shape

1 Introduction

Wordles [17] have proven to be simple but effective in conveying an overview of text in an engaging way [35]. As the size of a word represents its frequency (and thus weight), people can easily identify the main theme of the text. In addition, the Wordle algorithm produces an aesthetically pleasing and appealing output by leveraging typefaces and colors. As Wordles have been gaining a tremendous popularity, a wide variety of applications have been developed to enable people to create expressive and beautiful wordles (e.g., [12, 22, 23, 25, 37, 41]).

With the continuous popularity of Wordles on social media, a number of tools have emerged, such as WordArt [41] and Tagxedo [25], which allow people to create Wordles in a certain shape by fitting words into it. However, the results do not achieve a high fill rate and high data fidelity at the same time, as shown by two WordArt examples: the word sizes accurately encode the word frequencies at the cost of a loose filling in Fig. 2(a), while Fig. 2(b) achieves a tight filling by exaggerating the sizes of some words (e.g., Day and Church). In addition, we recognize the opportunity of facilitating the creation of semantically meaningful Wordles: associating topics into proper parts of a shape is helpful for presenting multiple topics, and can enhance the communicative power of Wordles [12, 37]. However, existing tools do not provide manipulation capabilities: (i) arrange words of different topics into different parts of a shape, (ii) directly manipulate individual words, and (iii) fine-tune the Wordle shape.

In this paper, we present *ShapeWordle*, a new technique that allows people to generate shape-bounded Wordles, while preserving data fidelity. The core of our technique is a shape-aware Archimedean spiral for guiding the word placement within a Wordle. Formulating the Archimedean spiral in a differential form enables us to guide it by a distance field, to generate spirals of arbitrary forms, and thus to fill arbitrary shapes with almost equidistant words. Figs. 2(c,d) show results generated by the original Wordle tool WordArt and our ShapeWordle for the same input text. In contrast to Fig. 2(d), the top and bottom por-

- *Y. Wang, X. Chu, K. Zhang, C. Bao, and X. Li are with Shandong University. Email: {cloudseawang, cuxiaoxie, sdukaiyi, baochen95, xiaotonglig}@gmail.com.*
- *J. Zhang is with CNIC, CAS. E-mail: zhangjian@sccas.cn.*
- *C.-W. Fu is with the Chinese University of Hong Kong. E-mail: cwfu@cse.cuhk.edu.hk.*
- *C. Hurter is with ENAC, France. E-mail: christophe.hurter@enac.fr.*
- *B. Lee is with Microsoft Research. E-mail: bongshin@microsoft.com.*
- *O. Deussen is with Konstanz University, Germany and Shenzhen VisuCA Key Lab, SIAT, China. E-mail: oliver.deussen@uni-konstanz.de.*
- *Y. Wang and X. Chu are joint first authors and J. Zhang is the corresponding author.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

Fig. 2. Comparing word clouds produced by different algorithms: generated by WordArt [41] by (a) using the input word frequency and by (b) further *"tweaking"* the actual word frequency to fill the space, without respecting the data fidelity; (c) generated by a traditional Wordle layout algorithm [35] but taking the shape as the boundary constraint; (d) generated by ShapeWordle, our algorithm; and (e) all words enlarged at the same rate for a dense filling using ShapeWordle (note that such operation is not supported by WordArt). Also, note that the colors do not encode any information; we manually assign consistent colors for the words to allow more convenient comparisons across the results.

tions in Fig. 2(c) are nearly empty due to the compact spiral placement strategy of Wordle. In addition, the word distribution in Fig. 2(d) is more uniform than the one in Fig. 2(a), which was produced using WordArt. Note that for a fair comparison we intentionally used the same set of words with exactly the same sizes for both approaches. To obtain a high fill rate, even when there are not enough words, ShapeWordle is able to uniformly scale all words to fill the shape, while preserving their relative weights (see the example shown in Fig. 2(e)).

Besides, ShapeWordle is able to generate multi-centric layouts by performing a greedy layout strategy to fill each part of a complex shape with its associated words. Meaningful shape parts are obtained by first using automatic image/shape segmentation methods [31] and then performing interactive fine-tuning.

It is challenging to allow interactive editing while achieving a high filling rate [37] in a word cloud. To tackle this challenge, we develop a hybrid word representation that allows the editing of important words in a cloud and filling additional (typically smaller) words in a subsequent step. The division between the two groups can be done by selecting a certain percentage of the most frequent words (those with the biggest size) or by selecting all words beyond a size threshold. Once the words are split and all important words have been arranged, users are allowed to interactively manipulate each editable word, change the overall shape and refine the correspondence between words and shape parts. Due to this editing functionality, ShapeWordle enables users to create semantically meaningful wordles such as two examples in Fig. 1, which cannot easily be achieved by existing tools.

We evaluated our approach by quantitatively measuring the quality of our results by computing the layout coverage [5], layout uniformity and shape similarity. The results show that our method is capable of producing compact Wordles with shapes highly similar to the given outlines. We also invited 14 people to investigate ShapeWordle as an authoring tool. Our results demonstrate the advantages of ShapeWordle over approaches such as WordArt.

In summary our main contributions are:

- We formulate a shape-aware Archimedean spiral to guide and align Wordle layouts with arbitrarily-given shapes and to facilitate us to create multi-centric Wordles, where different words are placed in different parts of the given shape;
- We introduce a set of shape-aware Wordle editing interactions based on the coherent combination of rigid body operations and pixel-based placements; and
- We quantitatively evaluated the quality of the resulting Wordles and conducted case studies with 14 users to illustrate the expressiveness of shapeWordle.

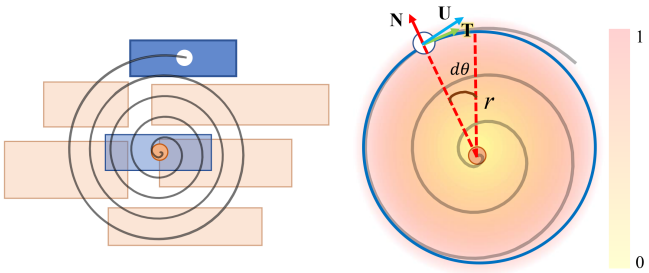
2 Related Work

2.1 Word Cloud Visualization

A complete review of the design space of word clouds is beyond the scope of this paper. We refer the readers to Felix et al. [18]. Here, we focus our discussion on the layout problem, i.e., given a set of words with associated weights, create a layout of words, say with one of the following three options: horizontal, vertical and spatial. The first two options simply arrange words from top to bottom or from left to right in alphabetical order or by their weights. In contrast, the last option does not impose any specific order of positioning the words, but arranges them subject to various aesthetic and semantic criteria. A classic example is Wordle [35], which attracts a large number of users in recent years. The core of a Wordle is a greedy algorithm that successively layouts words along a spiral. The original algorithm, however, does not meet many aesthetic and semantic needs of designers, since it does not allow users to manipulate individual words nor pack the words into a target shape. To this end, a variety of advanced layout and editing methods have been proposed in recent years.

Neighborhood graphs. To arrange relevant words next to each other for forming a semantic layout, often a neighborhood graph is constructed with nodes representing words and edges connecting relevant words with weights. Cui et al. [14] implicitly construct such a graph by creating a distance matrix that describes the cosine similarity between words, then place words via a multidimensional scaling of the matrix using a force-directed scheme to reduce the empty space between words. Wu et al. [42] improve this layout by using seam carving to further remove the empty space, while Paulovich et al. [28] extend this algorithm for visualizing the neighborhood relationship between documents and their corresponding word clouds simultaneously. Rather than using an implicit graph, Barth et al. [4] directly incorporate a neighborhood graph into their word clouds. They show that respecting the neighborhood relationship is an NP-hard problem. To overcome the related computational overhead, they present several approximation algorithms and conduct a quantitative comparison using implicit methods [14, 42], and show that Wordle is the most compact layout. In this paper, our focus is to extend Wordle and shape the Wordle into a target shape, while weakly respecting the neighbor graph.

Spatial information. Two kinds of spatial information have been integrated into word clouds: geo-spatial position of each word within a cloud and artistic shape creation. Buchin et al. [9] design geo word clouds that place words to respect not only the word frequency but also the relative positions between words. Since the spatial relations between words are preserved, such clouds can form shapes of geographic regions. WordArt [41] is a tool for generating compact word clouds in arbitrarily-specified shapes (see Fig. 2(b)). The results, however, do not respect the actual word frequency, or the data fidelity. Chi et al. [12] show an alternative way that takes a Wordle layout as input and employs constrained rigid body dynamics to re-arrange the words into a target shape. However, if the initial layout is not similar to the given shape (which is often the case), the rigid body dynamics often fail to re-shape the layout for the target shape. In contrast, our ShapeWordle directly layouts words in a given shape using a shape-aware Archimedean spiral, allowing the creation of compact layouts.



(a) (b)

Fig. 3. (a) Along the spiral started from the orange dot, search for a position to place the next word (blue rectangle at the white dot), such that the word does not overlap with the existing words. (b) Illustrating the spiral movement direction **U** and the normal (**N**) and tangent (**T**) directions at the white point on the circle (in blue) of radius r. The circle can be taken as the isoline of the underlying scalar field from the origin, where the color map on the right reveals the scalar field magnitude (r).

Temporal coherence. Temporal word clouds reveal temporal changes in a set of time-varying words, while preserving their temporal order. By combining parallel coordinates and traditional word clouds, parallel TagClouds [13] arrange words in each time step along one axis and show the word changes through connecting edges. SparkClouds [24] show trends between multiple word clouds by integrating sparklines into them. While both methods perform well for trend visualizations, they do not generate compact layouts. Alternatively, Cui et al. [14] create temporally-coherent word clouds using a multidimensional scaling and a force-directed model to layout words. Chi et al. [12] propose morphable word clouds [12] by taking an additional shape sequence as the input. They not only arrange words into a target shape but also preserve the temporal coherence of the word positions; see Fig. 15.

Interactive editing. Another form of interactions on word clouds is to allow the users to manually customize the cloud appearance. In the original Wordle tool, since only the global properties can be changed but not the individual words, Koh et al. introduce ManiWordle [23] that allows users to manipulate the typography, color, and composition of individual words. Later, Jo et al. [22] extend this method for multitouch editing. Although ManiWordle provides flexible controls, it may produce inconsistent and unpredictable layout changes due to its underlying placement strategy. To overcome this limitation, Wang et al. [37] propose EdWordle to preserve the neighbor relations between words in layouts during the user edits. Enabled by such capability, users can create semantic Wordles even in irregular shapes, although doing so requires highly tedious manual edits. Our ShapeWordle not only automatically generates word clouds that respect a target shape, but also provides rich interactions for users to edit the shapes and also the correspondences between words and parts in the shapes.

2.2 Spiral-based Visualization

Spirals are used in many spatial layout processes to compactly arrange objects. Among their many different forms [36], the Archimedean spiral is a widely-used one for visualization, since it is known to be effective in representing periodicity [1]. Already, Gabaglio [19] employed it to present periodic data. Carlis et al. [10] and Weber et al. [39] independently present the first prototypes of spiral displays, where the color and line thickness are used to encode time series data. Later, Dragicevic and Huot [15] combine spirals with a clock metaphor and develop a *SpiraClock* system for showing upcoming events. By arranging glyphs that encode multiple variables along a curve, a spiral can further be employed to reveal multivariate data [38], image-based search results [32], as well as network security data [6].

On the other hand, Archimedean spirals have been used as the underlying visual pattern to guide the placement of visual items. Two examples are Wordles [35] and balloon treemaps [34], where the words and circles are arranged along a spiral, starting from the origin using a greedy strategy. These examples, however, follow a conventional circular spiral, hence, they might not effectively fill an arbitrary target shape (see Fig. 2(c)). In this work, we generalize the Archimedean spirals to better adapt the word placement in target shapes. This allows us to optimize the generation of Wordles for arbitrary shapes.

3 Background

In this section, we first review the Wordle layout algorithm and discuss its two inherent drawbacks that limit its ability to shape a Wordle. After that, we briefly describe the Archimedean spiral formulation.

3.1 Wordle Layout Algorithm

Given a list of words and a weight associated with each word, the Wordle algorithm adjusts the size of each word in proportion to its weight and then represents the boundary of each word using a spline-based contour. To arrange the words in a compact and non-overlapping manner with the more important words closer to the centroid, the algorithm first sorts the words by the weights in descending order and then takes the following two steps to place one word at a time:

1. **Initialize:** pick a random position around the center of the canvas (see the orange dot in Fig. 3(a));
2. **Search-and-update:** create a spiral started from the picked random position, and search along the spiral for a location to place the next word, such that the next word does not overlap with any already-placed word; then, update the word cloud with the word placement (see Fig. 3(a) for an illustration of the process).

Being able to place words in horizontal, vertical or diagonal directions allows the creation of many variants. While the initial position can be completely random, the final Wordle might not be very compact.

Drawbacks. Both steps heavily limits the flexibility of Wordle in creating arbitrarily-shaped word clouds. First, the Archimedean spiral always searches for the new position in a circular manner, so the generated Wordle cannot effectively comply with the target shape (see again Fig. 2(c)). Second, picking the initial position around a center produces a single-piece Wordle, hindering the creation of multi-topic word clouds [37]. Overall, these two factors largely attribute to the general blobby shape of most Wordles. In contrast, storytelling word clouds convey semantics by arranging words into complex, multi-part shapes. There is a gap between the user demands and Wordle functionality.

3.2 Archimedean Spiral

The Archimedean spiral is one of most widely-used Euclidean spirals, which can be readily defined in polar coordinates:

$$r(\theta) = m\theta + b\,, \tag{1}$$

where θ is the polar angle, r is the radial distance from the origin, $b = r(0)$ is the initial distance of the starting point from the origin, and m controls the spacing between successive turns. Having a uniform spacing ($2m\pi$) between successive turns is an important and useful characteristic of the Archimedean spiral for many applications in medical imaging [27], material design [29], and digital light processing [3]. Such characteristic facilitates an efficient (uniform) space filling (see Fig. 3), enabling the creation of compact Wordles.

4 Shape-aware Wordle

In this section, we present how we achieve shape-aware Wordles by extending the Archimedean spiral to be shape-aware, and by supporting the generation of multi-centric Wordle layouts.

4.1 Shape-aware Archimedean Spirals

The Archimedean spiral can also be expressed in Cartesian coordinates, x and y, by using trigonometric functions:

$$\begin{pmatrix} x \\ y \end{pmatrix} = r(\theta) \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}\,. \tag{2}$$

Taking the derivatives of Eq. (2) with respect to θ yields

$$\begin{cases} \frac{dx}{d\theta} &= m\cos\theta - r(\theta)\sin\theta \\ \frac{dy}{d\theta} &= m\sin\theta + r(\theta)\cos\theta\,. \end{cases}$$

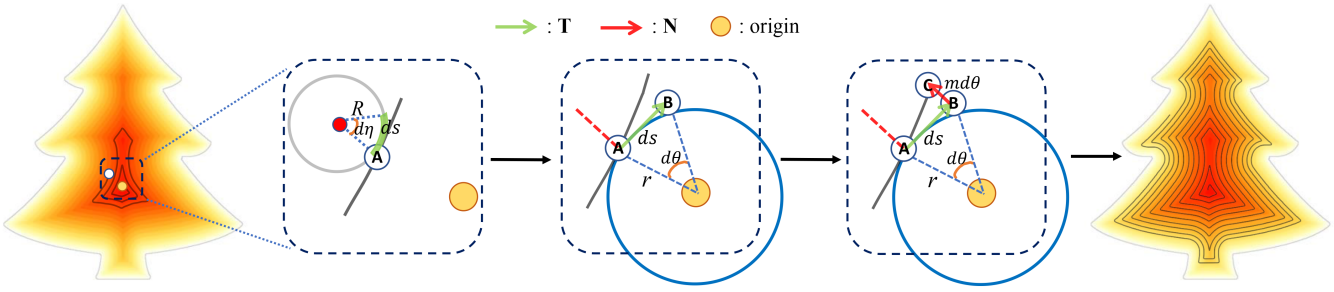


Fig. 4. Shape-aware Archimedean spirals. (a) Distance field created from a Christmas tree contour, with the spiral starting from the orange dot and currently stopping at the white dot; the zoomed view (b) shows the circle of curvature at the point A, where R is the local curvature radius at A and the red point marks the corresponding circle center. We approximate the movement distance from A to B (denoted as ds) by the arc length $Rd\eta$, where $d\eta$ is a user-specified parameter for angular speed. (c) Also, we approximate the length of ds by another arc $rd\theta$. (d) The distance from B to C is computed along the normal direction (in red) with length $md\theta$ (based on Eq. (4)). (e) Our generated shape-aware spiral for the Christmas tree.

Actually, $(\frac{dx}{d\theta}, \frac{dy}{d\theta})$ is the movement direction (denoted as $\mathbf{U}$) of the spiral at (x,y) in the 2D space (see Fig. 3(b) for an illustration). Here, we can decompose $\mathbf{U}$ along $\mathbf{N} = (\cos\theta, \sin\theta)^\mathsf{T}$ and $\mathbf{T} = (-\sin\theta, \cos\theta)^\mathsf{T}$:

$$\mathbf{U} = m\begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix} + r(\theta)\begin{pmatrix} -\sin\theta \\ \cos\theta \end{pmatrix} = m\mathbf{N} + r(\theta)\mathbf{T}\,, \qquad (3)$$

where $\mathbf{N}$ and $\mathbf{T}$ are the unit normal vector and unit tangent vector, resp., at point (x,y) on a circle of radius $\sqrt{x^2+y^2}$ co-centered with the spiral (see the blue circle in Fig. 3(b)). Such a circle can be interpreted as the isoline of an underlying distance field $\phi(x,y) = \sqrt{x^2+y^2}$, which measures the Euclidean distance from (x,y) to the origin.

According to Eq. (3), the movement direction $\mathbf{U}$ of the Archimedean spiral is governed by $\mathbf{N}$ and $\mathbf{T}$. To extend the spiral to be shape-aware with a *roughly constant spacing* between successive turns, we first compute the distance field (denoted by φ) associated with the input shape (see Fig. 4(a)). Then, we align $\mathbf{N}$ and $\mathbf{T}$ with the isolines of φ, instead of the isolines of the generic circular distance field ϕ shown in Fig. 3(b). In this way, when we construct the spiral from a starting point inside a shape, the spiral can move in a way that follows the shape and eventually meets the shape contour (see Fig. 4(d)).

Computing the distance field. A distance field is an effective shape representation that has been used for edge bundling [16] and trail data visualization [20]. It is a scalar field that specifies the shortest distance to a shape contour specified by a distance transform $\mathbb{R}^2 \to \mathbb{R}_+$:

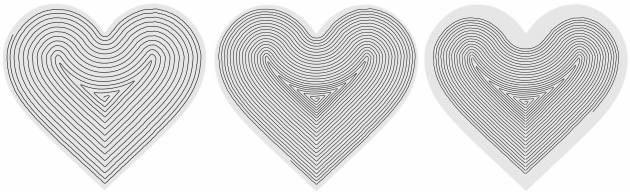
$$\varphi(\mathbf{p} \in \mathbb{R}^2, \Omega) = \min_{\mathbf{q}\in\Omega} ||\mathbf{p} - \mathbf{q}||\,,$$

where $\mathbf{p}$ is a point in 2D space, Ω is the shape contour, and $\mathbf{q}$ is any point on Ω. Note that φ is zero on the shape contour and gradually increases towards the center or the medial axis of the shape, and we compute the distance field using a linear algorithm [8] with time complexity $O(n)$, where n is the number of points in 2D space.

Extending the Archimedean spiral. To extend the Archimedean spiral to be shape-aware, the main question is how to guide the movement of the spiral, or how to define the movement direction of the spiral at any point $\mathbf{p}$ in the given shape. Rather than explicitly constructing the isoline and then computing the isoline normal at $\mathbf{p}$, we take the gradient of the distance field as $\mathbf{N}$. This strategy can accurately approximate the normal [33] for continuous scalar fields, like φ. Once $\mathbf{N}$ is available, $\mathbf{T}$ can be easily obtained because it is a unit vector that is orthogonal to $\mathbf{N}$. Then, we can re-write Eq. (3) in a differential form:

$$\begin{pmatrix} dx \\ dy \end{pmatrix} = md\theta\mathbf{N} + rd\theta\mathbf{T}\,, \qquad (4)$$

where $r = \sqrt{x^2+y^2}$. However, using the same θ at every point in an arbitrary shape might not be proper, since the generated spiral might not be able to adapt to regions of high-curvature, e.g., near the corners of the Christmas tree in Fig. 2.



(a)m = 1.5, $d\eta = \pi/5$, nIter = 15000 (b) m = 1, $d\eta = \pi/5$, nIter = 18000 (c) m = 1, $d\eta = \pi/10$, nIter = 18000

Fig. 5. Examples of shape-aware Archimedean spirals generated on the same shape using different parameters m, $d\eta$, and number of iterations (nIter).

To characterize such sharp features, we consider the local curvature along the spiral and approximate the curve by small tangential movements (denoted by ds) perpendicular to $\mathbf{N}$ by $Rd\eta$ and also by $rd\theta$ (see Fig. 4(b) and (c)), where R is the local curvature radius and η is a user-specified parameter for the angular speed. Doing so, we can write

$$d\theta = \frac{Rd\eta}{r}. \qquad (5)$$

Note that r in Eq. (4) can also be regarded as the local curvature radius defined on the isolines (concentric circles) of the generic distance field ϕ (see Fig. 4(c)), while R is the local curvature radius defined on the isolines of the shape-aware distance field φ. To estimate R at an arbitrary point in φ, we employ the Hessian matrix [7]; see the supplemental material for details. Putting everything together, we can rewrite Eq. (4) as

$$\begin{pmatrix} dx \\ dy \end{pmatrix} = \frac{mRd\eta}{r}\mathbf{N} + Rd\eta\mathbf{T}\,. \qquad (6)$$

Note that R and r are different at different points, while m and $d\eta$ are constant parameters specified by the user.

In our implementation, we compute dx and dy using Eq. (6) to progressively trace out the spiral from the origin. Typically, we set $m = 1$ and $d\eta = \pi/5$, while m and R are measured in pixel units on the shape image. Since m is a constant, the generated spirals can have nearly-uniform spacing ($2m\pi$) between successive turns. Fig. 5 shows some examples. In the case of a straight curve segment R might become too large, so we empirically set an upper bound of 1.5 pixel units as a maximum movement distance along the tangential direction to avoid excessive movements. We ran our method on a quad-core PC with a 23" LCD widescreen, an Intel(R) Core(TM) i7-6700K CPU and 16GB RAM. For a wordle of 60 words, our method finishes in less than 5s.

Uniform scaling. Putting our shape-aware Archimedean spiral into the Wordle layout algorithm enables us to efficiently generate shape-aware Wordles. Fig. 2(d) shows an example result. However, if the total word areas is far less than the total shape area, it is hard to create a

compact layout. Hence, we define a uniform scaling parameter for the words in an attempt to maximize the fill rate, where the relative weights between words are still preserved. Fig. 2(e) shows the result after we scale up the important words in Fig. 2(d) and regenerate the Wordle.

4.2 Multi-centric Layout

For an input shape with multiple components, we use a shape-aware Archimedean spiral to generate a Wordle layout for each of the components. Fig. 6 shows an example with three components, where we generate a multi-topic Wordle by filling words of a specific topic in each of the components: tulip, spring, and flower bulb.

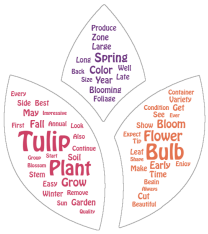


Fig. 6. A shape with three components.

However, if a component has multiple local maxima of the distance field, generating a Wordle layout for the whole component might not be able to fill it completely. The shape shown in Fig. 7(a) has two components. Tracing only a single spiral from the global maximum in the right component misses the top portion of the wing and results in a large empty area. To accommodate the Wordle algorithm for such non-convex components, we use a multi-centric layout to generate the spirals and place words around multiple local maxima.

Shape Segmentation. Given a shape, we detect connected components in the shape and generate a distance field per component. We then use an iterative gradient-descent procedure [11] to locate the local maxima(s) and the associated shape region(s) (called as parts) in each component. This allows us to implicitly segment a component into a few parts. Fig. 7(b) shows an example, where the two components of the pigeon shape are segmented into four parts.

Word Assignment. Given a list of words to fill a shape, we first set a font size for each word such that the sum of the areas of all the words is 70% of the total shape area. We then use a greedy strategy to assign words to the different parts of the shape. Denoting $p_{i,j}$ as the j-th part of the i-th component, $A_{i,j}$ as the area of $p_{i,j}$, and N as the total number of input words, the number of words to be assigned to $p_{i,j}$ is

$$\frac{A_{i,j}}{\sum_u \sum_v A_{u,v}} N \,. \tag{7}$$

Assuming that the word with the largest weight should be assigned to the largest part, we then define the largest weight of the words in each part as

$$w_{i,j} = \frac{A_{i,j}}{\max_{u,v} A_{u,v}}. \tag{8}$$

Once $n_{i,j}$ and $w_{i,j}$ have been determined, we simultaneously and randomly assign words from the input word list to every part, such that the weight of any word in $p_{i,j}$ should not exceed $w_{i,j}$. If we run out of space for word placements due to such an assignment, we uniformly shrink all the words, and repeat the word assignment process.

By tracing multiple spirals around local maxima per part instead of per component, it is possible to fill also non-convex shape with words; Examples are given in Fig. 7(a) versus (c). On the other hand, using traditional Archimedean spirals to fill parts cannot solve the problem; as demonstrated in Fig. 7(d), circular spirals cannot effectively fill the abdomen part of the pigeon, resulting in a large empty area.

5 Interactive ShapeWordle Creation System

In this section, we present our ShapeWordle system, which facilitates users to interactively create and edit arbitrarily-shaped word clouds, while preserving their aesthetic characteristics. Like existing word cloud editing systems [23] [37], our system gives user controls over the layout through interactive manipulation of individual words and shapes. On top of these, as presented in the pipeline shown in Fig. 9, the users can simply load a piece of text and load/select a shape (Fig. 9(a)), apply a shape segmentation (Fig. 9(b)), and click a button to generate a word cloud that completely fills the given shape (Fig. 9(c)).

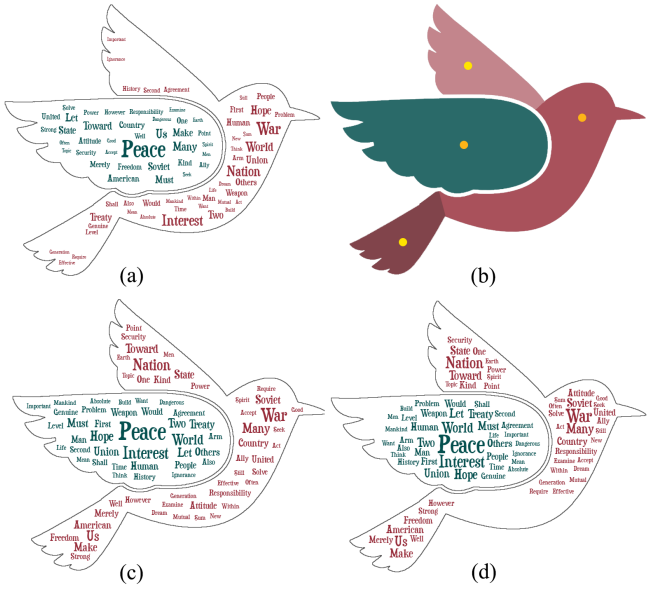


Fig. 7. Multi-centric layouts created by using multiple shape-aware Archimedean spirals: (a) tracing one spiral *per component* without shape segmentation; (b) the four parts segmented in the pigeon; (c) result after tracing one spiral *per segmented part*; and (d) result by tracing one spiral per part using the traditional Archimedean spirals instead.

Before the layout generation, the user can manually associate semantically-related words into specific segmented parts to create multi-topic Wordles. If the segmentation parts are not meaningful, he/she can refine them via interactive brushing; see an example in Fig. 8. Once a word cloud is generated, the user can further manipulate the important words with various editing operations (Figs. 9(d,e)) and fill in the less important words to produce the final Wordle (Fig. 9(f)). In the following, we describe how to completely fill a shape with a two-step layout and perform interactive word cloud editing.

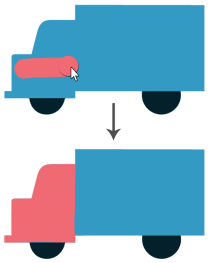


Fig. 8. Refining a segmentation by brushing.

5.1 Two-step Wordle Layout

As shown in Fig. 2(e), it might not be always possible to completely fill up a shape with large words only, even we uniformly scale all the words. On the other hand, users are mainly attracted by the large words when viewing a word cloud [2, 23], while paying less attention to the smaller ones. To efficiently fill a shape, we separate the input words into two classes: *core words* are the top N important words that can be interactively manipulated, while *marginal words* are used for filling up the space. To provide consistency-preserving word cloud editing [37], we represent the core words by their bounding boxes for layout, which are viewed as rigid bodies at the editing stage. In contrast with the core words, each marginal word is represented as a binary mask image [23] with "1" as foreground and "0" as background.

We use this combined representation to generate layouts in two steps, i.e., placing the core words then marginal words. First, we use our shape-aware Archimedean spirals to position the core words, so that their overall shape better aligns with the input shape. Note that if there are not enough core words, the layout might not be completely filled (see Fig. 2(d)). To clearly show the core words, we uniformly enlarge them until they dominate most of the shape area (see Fig. 2(e)). In contrast, the marginal words are placed by using the traditional Archimedean spirals for two reasons. First, the large number of tiny marginal words are mainly used for filling the small empty regions remaining in the shape; they need not align with the shape. Second, computing the shape-aware Archimedean spirals for a large number of tiny marginal words is time-consuming, and unnecessary.

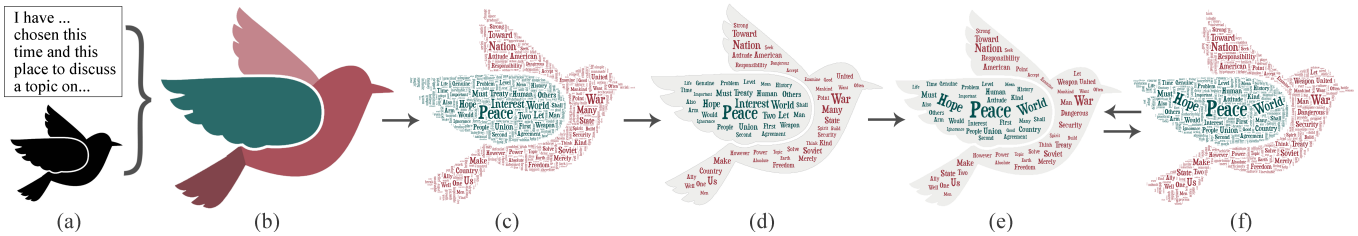


Fig. 9. Pipeline of our approach. (a) The input consists of a set of words with weight values and a shape; (b) the shape is segmented into different parts; (c) an initial Wordle filled up the given shape; (d) the layout of the important words in the editing mode; (e) the result generated after manipulating individual important words; (f) the result generated by filling the marginal words into the layout in (e) and if the further editing is required, user enters the editing model and the layout becomes the one shown in (e).

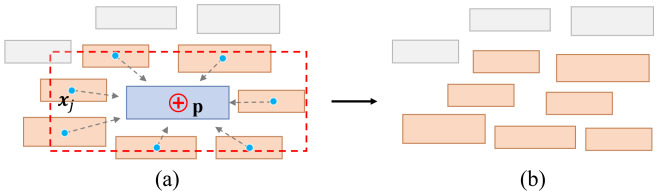


Fig. 10. Illustration of the uniform constraint: (a) central forces (grey arrows) towards the virtual rigid body in the middle, where the forces are applied to surrounding word boxes in the region marked by the red dashed box; and (b) updated layout, where the position of surrounding word boxes are re-adjusted.

5.2 Shaped Wordle Editing

After creating a shaped Wordle, users can manipulate it in the *editing* mode, where the marginal words disappear and users can modify the core words interactively (e.g., move, rotate, resize, delete), while preserving the neighborhood relationship between the core words. After the manipulation, the user can simply click a button to fill up the whole shape with marginal words. Below, we describe how we achieve consistent word cloud editing with rigid body dynamics [37].

In the editing mode, each word is boxed as a rigid body with a mass, and a customized rigid body dynamics [37] is applied to move the words by forces. The dynamics system allows us to avoid word overlap by imposing non-penetration constraints, which automatically detect collisions between bodies and move related bodies apart to respond to the collisions. In doing so, after the user moves/resizes a word, all surrounding related words will be automatically repositioned. To support intuitive shape-based editing, we customize the rigid body dynamics with two additional constraints:

(i) Boundary constraint. To keep the words inside their corresponding shape parts, a boundary constraint is applied to take the shape boundary pixels as static rigid bodies. If a word contacts with them, it will be bounced back. If the whole word (or a significant part) goes out of the boundary, we assume that the user intentionally takes the word out from the part, so we do not apply the boundary forces to the word.

(ii) Uniform constraint. Removing words will lead to large empty areas in the layout. To keep the layout uniform, we introduce a uniform constraint to insert virtual bodies to represent the empty areas with central forces to pull in the surrounding words.

Fig. 10 shows an example. Suppose a word is removed from (move out or delete) position $\mathbf{p}$, we insert a virtual rigid body at $\mathbf{p}$ and apply a pulling force on each surrounding word (say, at position $\mathbf{x}_j$):

$$F_j = \frac{\mathbf{v}_j}{||\mathbf{p}-\mathbf{x}_j||^2},$$

where $\mathbf{v}_j = \frac{\mathbf{p}-\mathbf{x}_j}{||\mathbf{p}-\mathbf{x}_j||}$ is the unit vector from $\mathbf{x}_j$ to $\mathbf{p}$. Since words that are further from the virtual body receive smaller forces, we set a range for applying the force to accelerate the computation. The range is defined by uniformly scaling the box of the original word (before removal) three times at the word center; see the red dashed box in Fig. 10(a) for an illustrative example and Fig. 10(b) for a result after applying pulling forces. Note that both constraints have been used for generating morphable word clouds [12], whereas here, they are only used in the editing mode. Incorporating these constraints might introduce flickering for words that collide with the static boundary, so we put them as options for users.

6 Evaluation

This section presents (i) a quantitative evaluation to compare ShapeWordle with the state-of-the-art, (ii) a case study, and (iii) how we extend ShapeWordle for creating temporally-shaped Wordles.

6.1 Quantitative Comparison

There are two existing tools we are aware of that can automatically generate shaped word clouds: WordArt [41] and Tagxedo [25]. For Tagxedo, we found that it might drop important words if it cannot accommodate the words in the layout, which is not faithful to the input data; see supplemental materials for visual comparisons with Tagxedo. Hence, we compare our ShapeWordle mainly with WordArt, which is the best tool, so far, for generating word clouds [26]. However, since the algorithm and code of WordArt are not accessible, we can only obtain outputs from WordArt in the form of regular RGB images. Hence, to compare with WordArt, we use ShapeWordle to generate regular RGB images in the same resolution as those from WordArt.

Metrics. We employ the following three metrics to quantitatively evaluate the quality of the generated word clouds:

- *Layout coverage* (**LC**) is extended from Barth et al. [4] for measuring the overall proportion of empty space in the generated layout. To do so, we count the number of pixels in text color (i.e., words in the clouds) and the number of pixels in non-text background color inside the shape. Hence, a large LC value closer to one means better coverage.
- *Layout uniformity* (**LU**). This metric, from another aspect, measures the distribution uniformity of the gaps among the words in the layout, meeting the basic requirements for wordles. We formulate LU based on the following observation: if the words in a layout are evenly distributed, the gaps in-between words should be small and similar in sizes. Hence, given a layout of words, we first generate a pixel-level distance field (denoted as φ), where each pixel stores the distance to the nearest pixel in text color; see Fig. 11(a) for an example. Thus, we define LU, which sums up the per-pixel squared distance values and normalizes the sum by the total number of non-text pixels inside the shape (n_{nontext}):

$$\text{LU} = \frac{1}{n_{\text{nontext}}} \sum_i \varphi(\mathbf{p}_i)^2 \,, \tag{9}$$

  where $\mathbf{p}_i$ is the i-th pixel in the shape. Here, a small LU indicates a better layout uniformity, and we square the distance values to penalize large distances (in pixel units).
- *Shape similarity* (**SS**). The third metric SS aims to measure how good the generated wordle aligns with the given shape. Here, we adopt the least distance model [21] to measure the distance from each boundary pixel. For each pixel on the shape contour Ω_B,

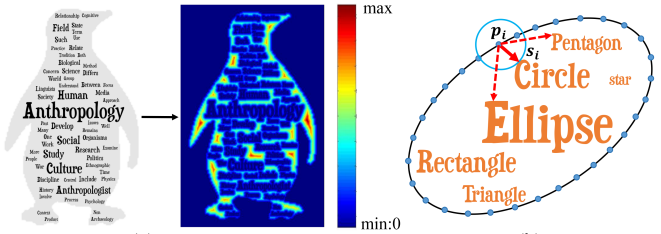


Fig. 11. Illustration: computing (a) layout uniformity using the distance field in input word cloud and (b) shape similarity over boundary pixel $\mathbf{p}_i$.

we find the distance (denoted as ρ) to the nearest text color pixel in the shape; see Fig 11(b) for an illustration. In this way, we can define SS by summing up per-pixel squared distances over the shape contour and normalizes the sum by the total number of pixels on the shape contour (n_{contour}):

$$\text{SS} = \frac{1}{n_{\text{contour}}} \sum_{\mathbf{p}_i \in \Omega_B} \rho(\mathbf{p}_i)^2 \,. \tag{10}$$

Similar to LU, a small SS indicates a layout that better matches the input shape contour.

Regarding the data fidelity, we did not explicitly include it as a metric, since our method already fully respects the original weight of words. For a fair comparison, we also intentionally set the WordArt to produce layouts that fully respect the data fidelity.

Data. We employed the dataset from Brath et al. [4], which includes 112 articles extracted from the English Wikipedia and 45 scientific papers from the proceedings of the conferences SEA and ALENEX. Also, we collected 157 shapes in diverse categories such as animals, plants, sports, and numbers, and then randomly assigned a unique shape to each document. To examine how our ShapeWordle behaves in single-centric and multi-centric Wordles, we further divided the shapes into two types: single-part shapes, e.g., the Christmas tree in Fig. 2, and multi-part shapes, e.g., the pigeon in Fig. 7.

Settings. Since the algorithm and code of WordArt are not available, we tried to use more consistent settings to compare ShapeWordle and WordArt. First, we draw all the words in both systems in horizontal orientation using the Duality font. Second, how WordArt maps word frequency (or weight) to scale the font sizes of the generated words is unknown. Hence, we first use WordArt to generate a word layout and then directly take the font size of the generated words (provided in a downloaded CSS file) as the input word weight in our ShapeWordle. Third, although both WordArt and ShapeWordle can completely fill a given shape with a large number of tiny words, to evaluate a layout quality, we consider mainly the layout of the large important words, which attract more user attention [30]. In addition, it is actually more challenging to use fewer words to more uniformly generate shaped word clouds [12]. Considering these, we fix the number of words as 60, which is often not enough to entirely fill the input shapes.

Results. We employed our method and WordArt to generate shaped word clouds for the whole dataset, and computed the LC, LU, and SS scores for each result. Screenshots of all the generated word clouds along with their LC, LU, and SS scores can be found in the supplemental material. Also, we created the boxplots shown in Fig. 12 to summarize the scores. In general, a large LC value indicates a better layout coverage, and it is not possible to achieve a value close to one, since LC counts only the text-color pixels in the shape. See also the LC values in the comparison results shown in Fig. 13. Here, we also want to highlight that for WordArt, we cannot uniformly scale all the words to improve the coverage, while in our case, thanks to the shape-aware Archimedean spirals, we can more evenly distribute the words, so we can further enlarge the words in our ShapeWordle results without word collisions. So, we can further improve the layout coverage.

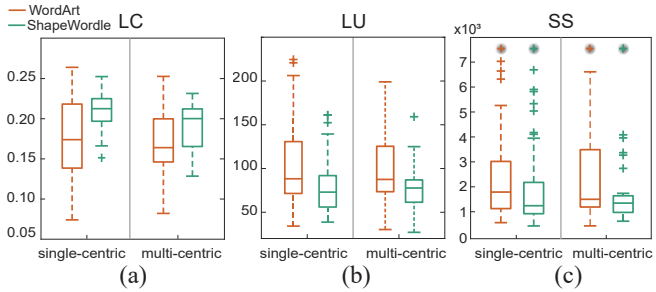


Fig. 12. Boxplots summarize the scores of LC, LU, and SS of our ShapeWordle (green ones) and WordArt (orange ones). The outliers with the dark halos are out of the plot range. A larger LC value and smaller LU and SS values indicate a better result.

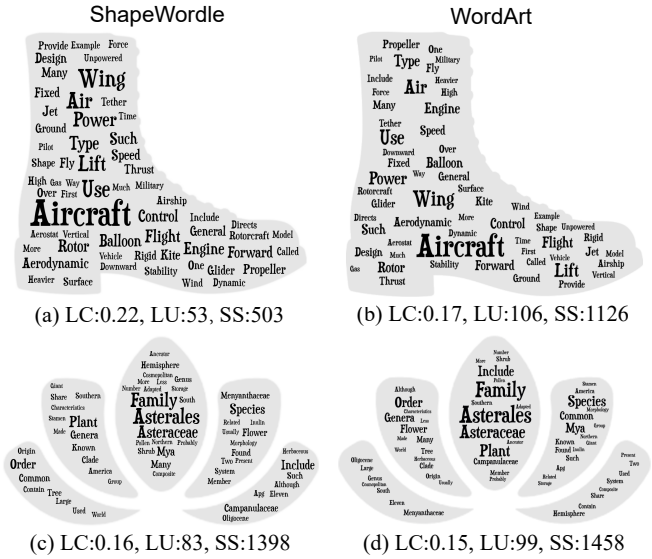


Fig. 13. Typical results produced from our ShapeWordle (left) and WordArt (right) for single-centric (top) and multi-centric (bottom) input shapes.

On the other hand, LU and SS measure pixel-level squared distances to account for the layout uniformity and shape similarity. A small LU value indicates better layout uniformity with smaller and more evenly-sized gaps in-between words in the results, while a small SS value indicates that the words can better match the shape contour. Thanks again to the shape-aware Archimedean spirals, which enable us to generate shape-aware layouts that are more uniform. Hence, we can distribute words more evenly in the shape, as demonstrated by the results shown in Fig. 13 and by the LU values. Also, ShapeWordle achieves lower SS values, since our spirals better fit the given shapes.

Furthermore, comparing the results for single- and multi-centric shapes, we can see that all the LC, LU, and SS scores drop, for both ShapeWordle and WordArt. This is reasonable, since multi-centric layouts typically contain multiple smaller parts (Fig. 13 (bottom)), where it is harder to fill in words that better match the individual parts. Yet, ShapeWordle still produces more compact and uniformly-distributed words for both single-centric and multi-centric shapes.

6.2 Case Studies

To explore the expressiveness of ShapeWordle, we performed case studies with 14 participants from different schools in a local university. During this study, we exposed all functions of ShapeWordle to the participants and asked them to create a layout of their own favors. We closely observed their actions, and after they were done, we had a short interview with each of them to collect their design philosophy and feedback on our system. Each case took about 5-20 minutes, including the design process and interactions. All the participants reported the creation process as engaging. *They agreed that the system was "easy to learn and fun to play with". Several users especially mentioned*

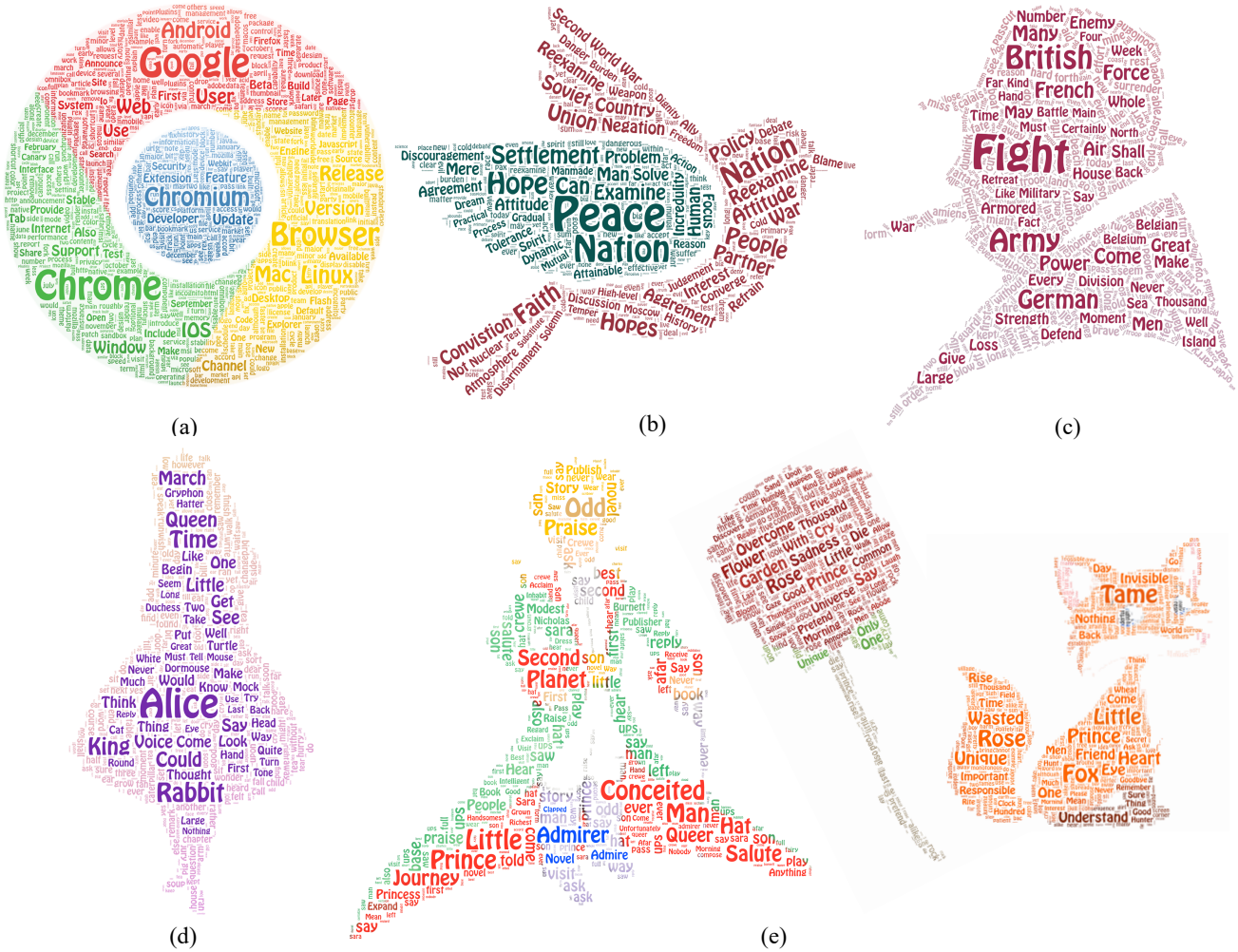


Fig. 14. Results designed by the participants in our case studies. (a) the Wikipedia page of Google Chrome [40]; (b) The speech "Towards a Strategy of Peace" by the former U.S. President John F. Kennedy; (c) the speech "We shall fight on the beaches" by the British politician Winston Churchill; (d) the novel "Alice's Adventures in Wonderland"; and (e) prince, rose, and fox from the novel "Little Prince."

they liked the friendly editing interactions, which allow them to freely move the words. Two of our participants have produced shaped word clouds using other tools before. They reported that when generating personalized layouts, our system saves their time and produces more satisfying results. Figs. 1 and 14 show several inspiring works produced by them; more results can be found in the supplemental material.

In the following, we briefly discuss some of the results:

VIS (Fig. 1) was created by an expert (non-author) in visualization. It includes two shaped Wordles. The left one has three parts: "V", "I", and "S", where the expert assigned words extracted from the abstracts of VAST, InfoVis, and SciVis papers of IEEE VIS 2018 to the three parts, respectively, followed by some refinements to relocate the words in three minutes. Note that since "I" is too thin, the word "visualization" is shortened as "Visual" to fit the shape. From this result, we can see the core topics in VAST, InfoVis, and SciVis, e.g., "Ensemble" for Scivis, "Set" for InfoVis, and "Cluster" for VAST, while the three parts also share some common topics like "Network" in InfoVis and VAST, and "Data" in all of them. For the right Wordle, since IEEE VIS 2019 will take place in Canada, the expert filled the words from the call for papers of IEEE VIS 2019 into the shape of the Canadian Maple Leaf, in which the relationship between Canada and IEEE VIS 2019 can be clearly shown. To make the layout more interesting, the expert intentionally edited the word positions over the leaf boundary.

Chrome (Fig. 14(a)) was created by a web programmer who took the Wikipedia page of "Google Chrome" as the data and directly used the colors in the Chrome logo. The most attractive words are "Google", "Chrome", and "Browser", each centering in its associated part. These three words form a triangle, indicating the main contents of this article. The word "Chromium" was put at the center of the layout because most of the source code came from the Chromium project. The words "developer" and "update" were also placed near the layout center. The designer wanted to indicate that this project was still being updated and attracts many developers around the world. The various operating systems where Chrome can work with were distributed in different parts of the logo to make the layout more uniform and balanced.

Pigeon (Fig. 14(b)) was designed by a student with web design experience, using a speech delivered by the former U.S. President, John F. Kennedy, "Towards a Strategy of Peace," in 1963. The speech has four parts. The first part was arranged on the pigeon's larger wing (in green), where the president appealed to people to examine their attitude toward peace. Here, the designer intentionally put the main word "Peace" in the middle and align some medium-size words with the feather. The second part was placed on the top wing, where the president talked about reexamining attitude toward the Soviet Union. The head and body of the pigeon were assigned with the third part: reexamine the attitude toward the cold war. Finally, the two important decisions that Kennedy mentioned last were placed on the tail. Overall, the designer spent around 20 minutes to create this result.

Winston Churchill (Fig. 14(c)) came from the famous speech of Winston Churchill, "We shall fight on the beaches." The participant took an image of Winston Churchill as the shape, and segmented the shape into three parts: head, face, and collar. Here, she deliberately re-positioned some large words, where she put "Fight" around the eye location because that is the theme of the speech, "British" in the head of the

character to indicate Churchill always kept his country and people in mind, and "Army" and "Power" in the throat of the character to show that these are the important elements during the war.

Alice (Fig. 14(d)) used the novel "Alice's Adventures in Wonderland" as the input data. Comparing to other cases, this result was done in just four minutes without much additional refinements. Overall, the participant used the outline of the main character in the novel Alice as the input shape and then enlarged some words that describe the major characters in the novel, such as Alice, King, Rabbit, and Queen. She commented that *"the big words align well with the shape. This saves me much time in editing because it's already so nice."*

Little prince (Fig. 14(e)) was created by a student interested in literature, where the data came from three different chapters in "Little Prince" for filling in the three separate shapes shown in the result, for prince, rose, and fox, from left to right. The first chosen chapter is about the little prince meeting a conceited man in the second planet, where the participant used the shape of the little prince for filling in words. Interestingly, the participant segmented the shape further into several parts for placing different topics of words, and he put some summary words such as 'Conceited" and "Queer" at the bottom part, etc. The second chosen chapter is about the rose. The participant emphasized several words that express feelings, e.g., "Overcome", "Cry", and "Sadness." The third chosen chapter is about the fox, where the participant picked the story about the little prince meeting the fox and becoming friend with her. The largest word in the head of the fox is "Tame," which is the main thing that the fox told the little prince. And in the tail of the fox shape, the participant intentionally expanded the words "wasted" and "rose" to highlight that *"it is the time you have wasted for your rose that makes your rose so important."*

6.3 An Extension for Temporal Data

Last, we extend our method to generate temporal shape Wordles with time-varying text data. Morphable word clouds [12] is the state-of-the-art for producing temporal shape wordles but it needs to solve a complex rigid body dynamics system. Thanks to the shape-aware Archimedean spirals, we can achieve the goal with a simpler strategy. Given a set of shapes over time, we first prepare correspondence between successive shapes. Hence, after we generate a layout for the first shape, we can take the layout as a reference to guide the word placement in the next shape. In this way, we can produce shaped Wordles over time, while accounting for the temporal coherency.

Fig. 15 compares the results generated by our method and Morphable word clouds [12] by using three key frames of the human life cycle data. We can see that our results (see Fig. 15(a)) are more compact and better highlight the large words, because of the background of filling words. See also the supplemental material for more comprehensive comparison.

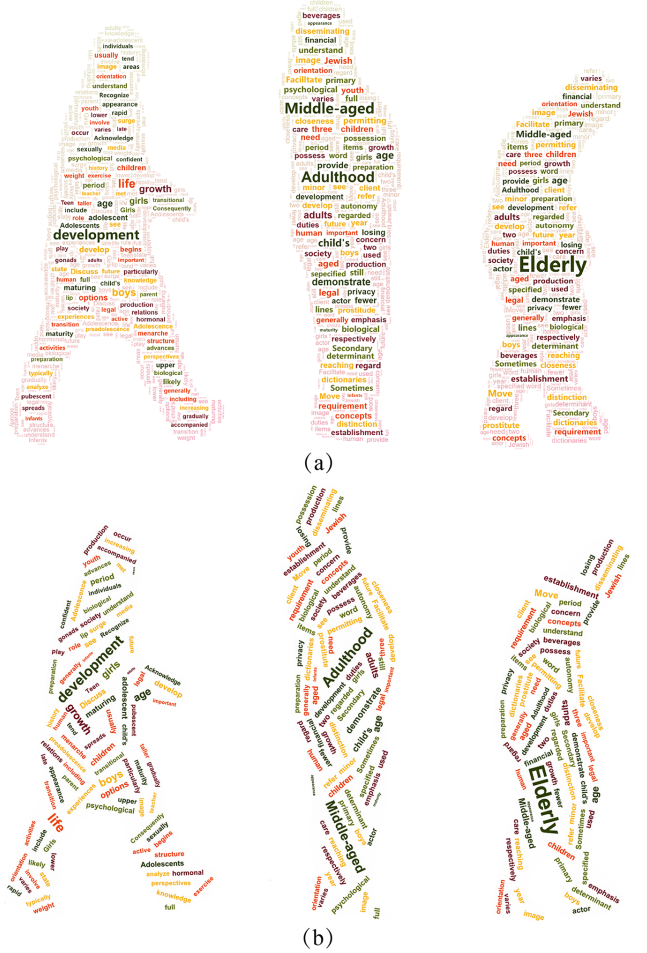


Fig. 15. The human life cycle from teenager to elder: (a) the three key frames created by our method; and (b) presented in Morphable word clouds [12]. Note that we assigned the same color to the same word in (a) and (b) to facilitate word-by-word comparison.

7 Conclusion

We presented ShapeWordle, a technique that facilitates users to create arbitrarily-shaped Wordles. The core of the technique is the shape-aware Archimedean spirals for guiding the word placement within a Wordle. By formulating the Archimedean spirals in a differential form, we can use the distance field of the shape to generate spirals in arbitrary forms, and accordingly produce shape-aware Wordles. For complex shapes, we introduce a multi-centric Wordle layout technique, which first segments the shape into parts then adaptively places words in each part by using shape-aware spirals. To allow users interactively editing the words, while densely filling the shape, we further develop a hybrid word representation that enables the editing of important words in the clouds and filling in additional small words. Further, we offer a set of editing interactions to facilitate the creation of semantically-meaningful Wordles. We evaluated with our system by quantitatively comparing it with the state-of-the-art tool, conducted case studies, and presented an extension for generating temporal word clouds.

There are still some limitations of our ShapeWordle that we will readily consider in the future work. First, the compactness and uniformity of our current multi-centric Wordles are not as good as those of single-centric layout; we will explore strategies to further improve our word assignments. Second, while our current method can place words in a shape-aware manner, it cannot automatically change the word orientation and align the words with the shape. For example, aligning the words in the feature of the pigeon shown in Fig. 14(b) took the participant around seven minutes. We will investigate the possibility of incorporating shape orientation into our spirals in the future. Last, we will explore more applications of our shape-aware Archimedean spirals in visualization.

Acknowledgments

This work is supported by the grants of the National Key Research & Development Plan of China (2016YFB1001404), NSFC (61772315, 61861136012), the Leading Talents of Guangdong Program (00201509), the CAS grant (GJHZ1862), the DFG Center of Excellence 2117 "Centre for the advanced Study of Collective Behaviour" (ID: 422037984), DFG Project 493/19 "Perception-based Information Visualization," and the Research Grants Council of the Hong Kong Special Administrative Region (Project no. CUHK 14203416).

References

[1] W. Aigner, S. Miksch, W. Müller, H. Schumann, and C. Tominski. Visual methods for analyzing time-oriented data. *IEEE Trans. Vis. & Comp. Graphics*, 14(1):47–60, 2008. doi: 10.1109/TVCG.2007.70415

[2] E. Alexander, C.-C. Chang, M. Shimabukuro, S. Franconeri, C. Collins, and M. Gleicher. The biasing effect of word length in font size encodings. In *Poster Compendium of the IEEE Conference on Information Visualization*, 2016.

[3] G. Ballou. *Handbook for sound engineers*. Focal Press, 2015.

[4] L. Barth, S. I. Fabrikant, S. G. Kobourov, A. Lubiw, M. Nöllenburg, Y. Okamoto, S. Pupyrev, C. Squarcella, T. Ueckerdt, and A. Wolff. Semantic word cloud representations: Hardness and approximation algorithms. In *Latin American Symposium on Theoretical Informatics*, pp. 514–525, 2014. doi: 10.1007/978-3-642-54423-1_45

[5] L. Barth, S. G. Kobourov, and S. Pupyrev. Experimental comparison of semantic word clouds. In *Int. Symp. on Experimental Algorithms*, pp. 247–258, 2014. doi: 10.1007/978-3-319-07959-2_21

[6] E. Bertini, P. Hertzog, and D. Lalanne. Spiralview: towards security policies assessment through visual correlation of network resources with evolution of alarms. In *Proc. IEEE Symposium on Visual Analytics Science and Technology*, pp. 139–146. IEEE, 2007.

[7] K. Binmore and J. Davies. *Calculus: concepts and methods*. Cambridge University Press, 2002.

[8] H. Breu, J. Gil, D. Kirkpatrick, and M. Werman. Linear time euclidean distance transform algorithms. *IEEE Trans. Pat. Ana. & Mach. Int.*, 17(5):529–533, 1995. doi: 10.1109/34.391389

[9] K. Buchin, D. Creemers, A. Lazzarotto, B. Speckmann, and J. Wulms. Geo word clouds. In *Proc. IEEE Pacific Visualization Symposium*, pp. 144–151, 2016. doi: 10.1109/PACIFICVIS.2016.7465262

[10] J. V. Carlis and J. A. Konstan. Interactive visualization of serial periodic data. In *Proc. ACM Symposium on User Interface Software and Technology*, pp. 29–38, 1998.

[11] Y. Cheng. Mean shift, mode seeking, and clustering. 17(8):790–799, 1995.

[12] 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 Trans. Vis. & Comp. Graphics*, 21(12):1415–1426, 2015. doi: 10.1109/TVCG.2015.2440241

[13] C. Collins, F. B. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Proc. IEEE Symposium on Visual Analytics Science and Technology*, pp. 91–98, 2009. doi: 10.1109/VAST.2009.5333443

[14] W. Cui, Y. Wu, S. Liu, F. Wei, M. X. Zhou, and H. Qu. Context preserving dynamic word cloud visualization. In *Proc. IEEE Pacific Visualization Symposium*, pp. 121–128, 2010. doi: 10.1109/PACIFICVIS.2010.5429600

[15] P. Dragicevic and S. Huot. SpiraClock: a continuous and non-intrusive display for upcoming events. In *Proc. Extended Abstracts on Human Factors in Computing Systems*, pp. 604–605, 2002.

[16] O. Ersoy, C. Hurter, F. Paulovich, G. Cantareiro, and A. Telea. Skeleton-based edge bundling for graph visualization. *IEEE Trans. Vis. & Comp. Graphics*, 17(12):2364–2373, 2011.

[17] J. Feinberg. Wordle-beautiful word clouds. `http://www.wordle.net/`. 2009.

[18] C. Felix, S. Franconeri, and E. Bertini. Taking word clouds apart: An empirical investigation of the design space for keyword summaries. *IEEE Trans. Vis. & Comp. Graphics*, 24(1):657–666, 2018. doi: 10.1109/TVCG.2017.2746018

[19] A. Gabaglio. *Teoria generale della statistica*, vol. 1. U. Hoepli, 1888.

[20] C. Hurter, A. Telea, and O. Ersoy. Moleview: An attribute and structure-based semantic lens for large element-based plots. *IEEE Trans. Vis. & Comp. Graphics*, 17(12):2600–2609, 2011.

[21] S. Jayanti, Y. Kalyanaraman, N. Iyer, and K. Ramani. Developing an engineering shape benchmark for CAD models. *Computer-Aided Design*, 38(9):939–953, 2006.

[22] J. Jo, B. Lee, and J. Seo. WordlePlus: Expanding Wordle's use through natural interaction and animation. *IEEE Computer Graphics and Applications*, 35(6):20–28, 2015. doi: 10.1109/MCG.2015.113

[23] K. Koh, B. Lee, B. Kim, and J. Seo. ManiWordle: Providing flexible control over Wordle. *IEEE Trans. Vis. & Comp. Graphics*, 16(6):1190–1197, 2010. doi: 10.1109/TVCG.2010.175

[24] B. Lee, N. H. Riche, A. K. Karlson, and S. Carpendale. SparkClouds: Visualizing trends in tag clouds. *IEEE Trans. Vis. & Comp. Graphics*, 16(6):1182–1189, 2010. doi: 10.1109/TVCG.2010.194

[25] H. Leung. Tagxedo website. `http://www.tagxedo.com/`. last visited 06/2017.

[26] M. McGee. The 9 best word cloud generators. `https://blog.polleverywhere.com/best-word-cloud-generator/`.

[27] L. Nanni, A. Lumini, and S. Brahnam. Local binary patterns variants as texture descriptors for medical image analysis. *Artificial intelligence in medicine*, 49(2):117–125, 2010.

[28] F. V. Paulovich, F. Toledo, G. P. Telles, R. Minghim, and L. G. Nonato. Semantic wordification of document collections. *Computer Graphics Forum*, 31(3pt3):1145–1153, 2012. doi: 10.1111/j.1467-8659.2012.03107.x

[29] N. Rahman and M. N. Afsar. A novel modified archimedean polygonal spiral antenna. *IEEE Transactions on Antennas and Propagation*, 61(1):54–61, 2013.

[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*, pp. 995–998, 2007. doi: 10.1145/1240624.1240775

[31] A. Shamir. A survey on mesh segmentation techniques. *Computer Graphics Forum*, 27(6):1539–1556, 2008. doi: 10.1111/j.1467-8659.2007.01103.x

[32] A. Spoerri. RankSpiral: Toward enhancing search results visualizations. In *Proc. IEEE Information Visualization Symposium*, pp. p18–p18. IEEE, 2004. doi: 10.1109/INFVIS.2004.56

[33] A. Van Gelder and K. Kim. Direct volume rendering with shading via three-dimensional textures. In *Proceedings of 1996 Symposium on Volume Visualization*, pp. 23–30. IEEE, 1996.

[34] F. Viégas, M. Wattenberg, J. Hebert, G. Borggaard, A. Cichowlas, J. Feinberg, J. Orwant, and C. Wren. Google+ ripples: A native visualization of information flow. In *Proc. Int. Conf. on World Wide Web*, pp. 1389–1398. ACM, 2013.

[35] F. B. Viegas, M. Wattenberg, and J. Feinberg. Participatory visualization with wordle. *IEEE Trans. Vis. & Comp. Graphics*, 15(6):1137–1144, 2009. doi: 10.1109/TVCG.2009.171

[36] D. H. Von Seggern. *Practical Handbook of Curve Design and Generation*. CRC Press, 1994.

[37] Y. Wang, X. Chu, C. Bao, L. Zhu, O. Deussen, B. Chen, and M. Sedlmair. EdWordle: Consistency-preserving word cloud editing. *IEEE Trans. Vis. & Comp. Graphics*, 24(1):647–656, 2018. doi: 10.1109/TVCG.2017.2745859

[38] M. O. Ward and B. N. Lipchak. A visualization tool for exploratory analysis of cyclic multivariate data. *Metrika*, 51(1):27–37, 2000.

[39] M. Weber, M. Alexa, and W. Müller. Visualizing time-series on spirals. In *Proc. IEEE Information Visualization Symposium*, vol. 1, pp. 7–14, 2001.

[40] Wikipedia. Google chrome — wikipedia, the free encyclopedia, 2019. "`https://en.wikipedia.org/wiki/Google_Chrome` [Online; accessed 18-June-2019].

[41] WordArt.com. WordArt website. `http://www.wordart.com/`. [Online; accessed 25-January-2019].

[42] Y. Wu, T. Provan, F. Wei, S. Liu, and K.-L. Ma. Semantic-preserving word clouds by seam carving. *Computer Graphics Forum*, 30(3):741–750, 2011. doi: 10.1111/j.1467-8659.2011.01923.x